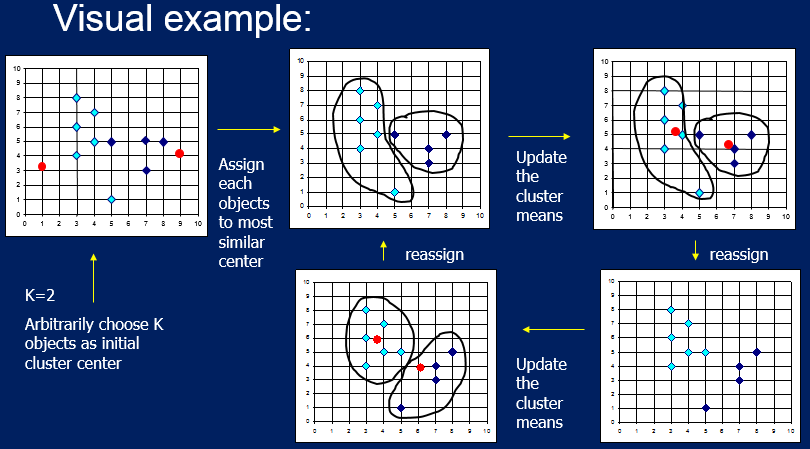
**5.4 - Cluster Analysis Using Partitioning Methods**

Partitioning methods use a different approach to form clusters. Given a choice for the number of clusters to be formed we start by randomly choosing k points in *p*-dimensional space to represent the cluster centers or *centroids*. Points are then classified to the cluster for whose centroid they are closest to. Then the cluster centroids are recomputed and the process is repeated until convergence, (i.e. no more changes can be made to the cluster assignments). The two methods we will examine are *k*-means and partitioning around medoids (PAM).

**K-Means Clustering**

The algorithm for *k*-means is summarized below for a given number of clusters *k*:

1. Partition objects to be clustered into *k* nonempty subsets.
2. Compute seed points as the centroids of the clusters of the current partition (the centroid is the mean vector for the observations in the cluster).
3. Assign each object to the cluster with the nearest seed point
4. Go back to Step 2, stop when no more new assignments.



Strength and Weaknesses of K-Means Clustering

Strength:

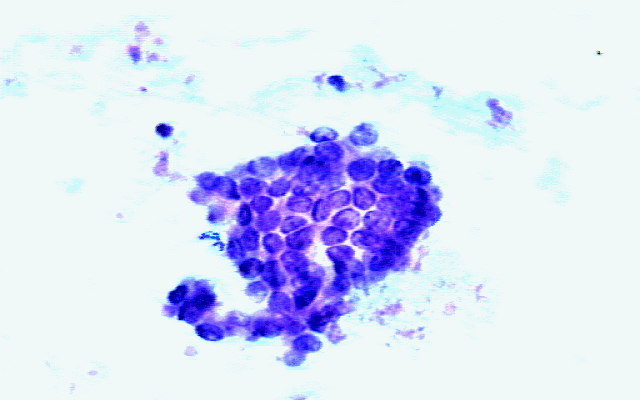
* Relatively efficient, does not require too many iterations to converge, though it often terminates at a local optimum and does not find the global optimum. Sometimes need to run it a few times.

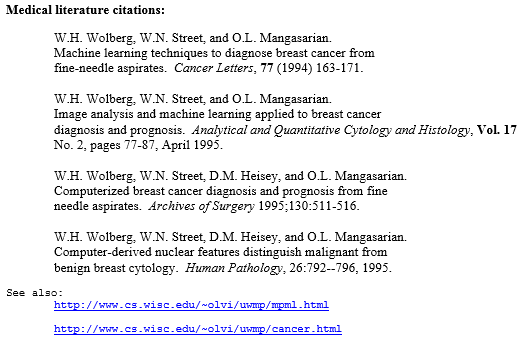
Weaknesses:

* Applicable only when mean is defined, then what about categorical data?
* Need to specify *k*, the number of clusters, in advance.
* Unable to handle noisy data and outliers.
* Not suitable to discover clusters with non-convex shapes.

Example 5.5: Breast Cancer Diagnosis Using Fine Needle Aspiration  
  
 Data Description:

Fine Needle Aspiration (FNA) is a fairly non-invasive method for sampling cells from a  
 breast tumor and then examining them using an electron microscope. The size and shape   
 features of cell can potentially effectively be used to classify a tumor as malignant or   
 benign. Features are computed from a digitized image of a fine needle aspirate (FNA) of   
 a breast mass. They describe characteristics of the cell nuclei present in the image. A   
 sample image is shown below.





Ten numeric features are the mean value based on three cells for the following cell   
 features:

* Radius = radius(mean of distances from center to points on the perimeter)
* Texture texture (standard deviation of gray-scale values)
* Perimeter = perimeter of the cell nucleus
* Area = area of the cell nucleus
* Smoothness = smoothness (local variation in radius lengths)
* Compactness = compactness (perimeter^2 / area - 1.0)
* Concavity = concavity (severity of concave portions of the contour)
* Concavepts = concave points (number of concave portions of the contour)
* Symmetry = symmetry (measure of symmetry of the cell nucleus)
* FracDim = fractal dimension ("coastline approximation" - 1)

The full data set contains the standard errors of the cell measurements (e.g. serad is the   
 standard error based on the three cell radius measurements) and worst case (maximum) value for   
 each (e.g. wrad = maximum cell radius of the three cells sampled)

> names(BreastDiag)

[1] "Id" "Diagnosis" "Radius" "Texture" "Perimeter" "Area"

[7] "Smoothness" "Compactness" "Concavity" "ConcavePts" "Symmetry" "FracDim"

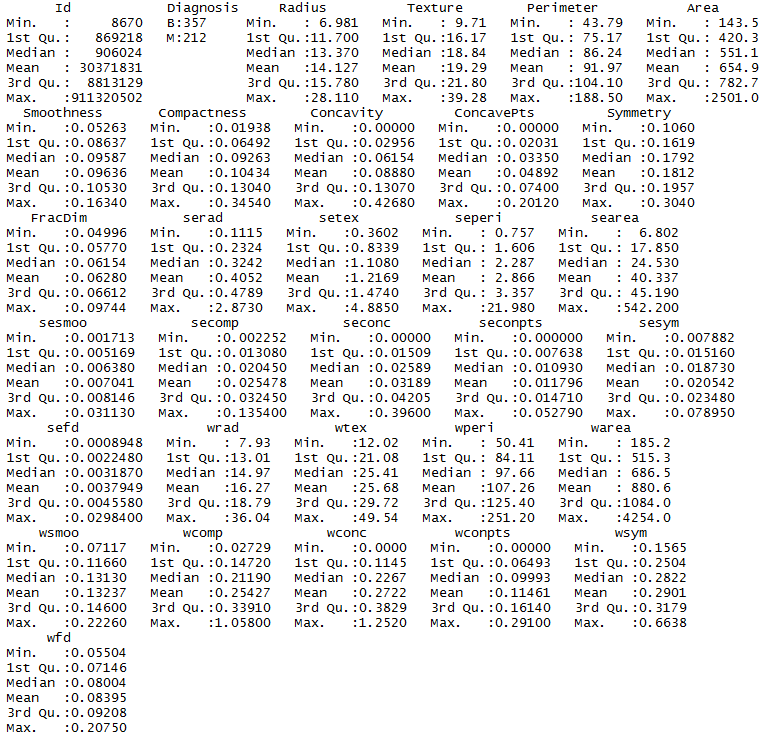
[13] "serad" "setex" "seperi" "searea" "sesmoo" "secomp"

[19] "seconc" "seconpts" "sesym" "sefd" "wrad" "wtex"

[25] "wperi" "warea" "wsmoo" "wcomp" "wconc" "wconpts"

[31] "wsym" "wfd"

> summary(BreastDiag)



> breast.mat = BreastDiag[,3:32]

> breast.mat = scale(breast.mat)

> breast.kmeans = kmeans(breast.mat,2)

> summary(breast.kmeans)

Length Class Mode

cluster 569 -none- numeric

centers 60 -none- numeric

totss 1 -none- numeric

withinss 2 -none- numeric

tot.withinss 1 -none- numeric

betweenss 1 -none- numeric

size 2 -none- numeric

iter 1 -none- numeric

ifault 1 -none- numeric

> table(breast.kmeans$cluster,BreastDiag$Diagnosis)

B M

1 343 37

2 14 175

Comments:

> library(FactoMineR)

> names(BreastDiag)

[1] "Diagnosis" "Radius" "Texture" "Perimeter" "Area"

[6] "Smoothness" "Compactness" "Concavity" "ConcavePts" "Symmetry"

[11] "FracDim" "serad" "setex" "seperi" "searea"

[16] "sesmoo" "secomp" "seconc" "seconpts" "sesym"

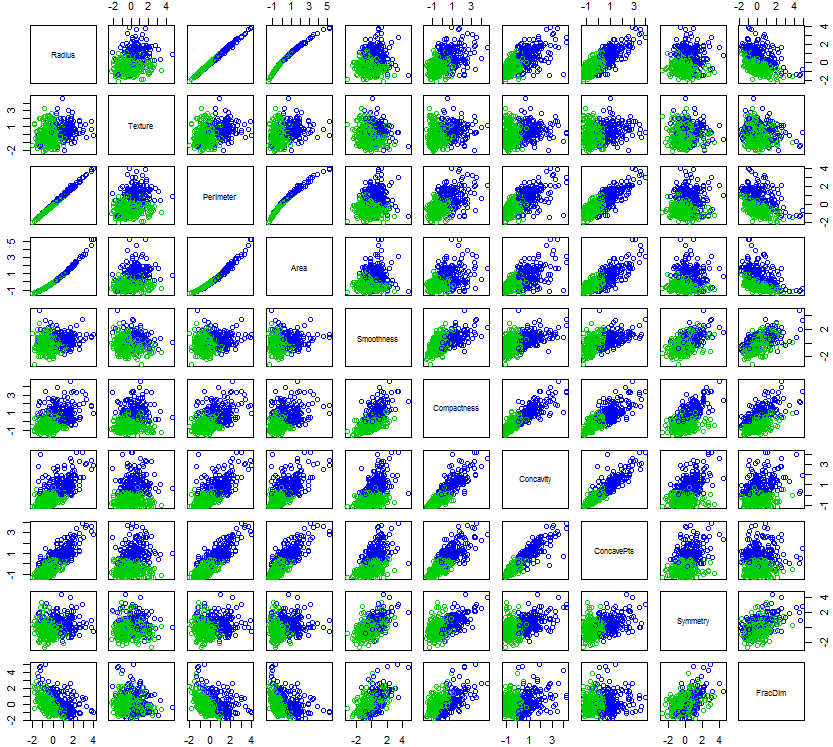
[21] "sefd" "wrad" "wtex" "wperi" "warea"

[26] "wsmoo" "wcomp" "wconc" "wconpts" "wsym"

[31] "wfd"

> breast.kmeans$withinss

[1] 4971.437 6603.711

> pairs(breast.mat[,1:10],col=breast.kmeans$cluster+2)  


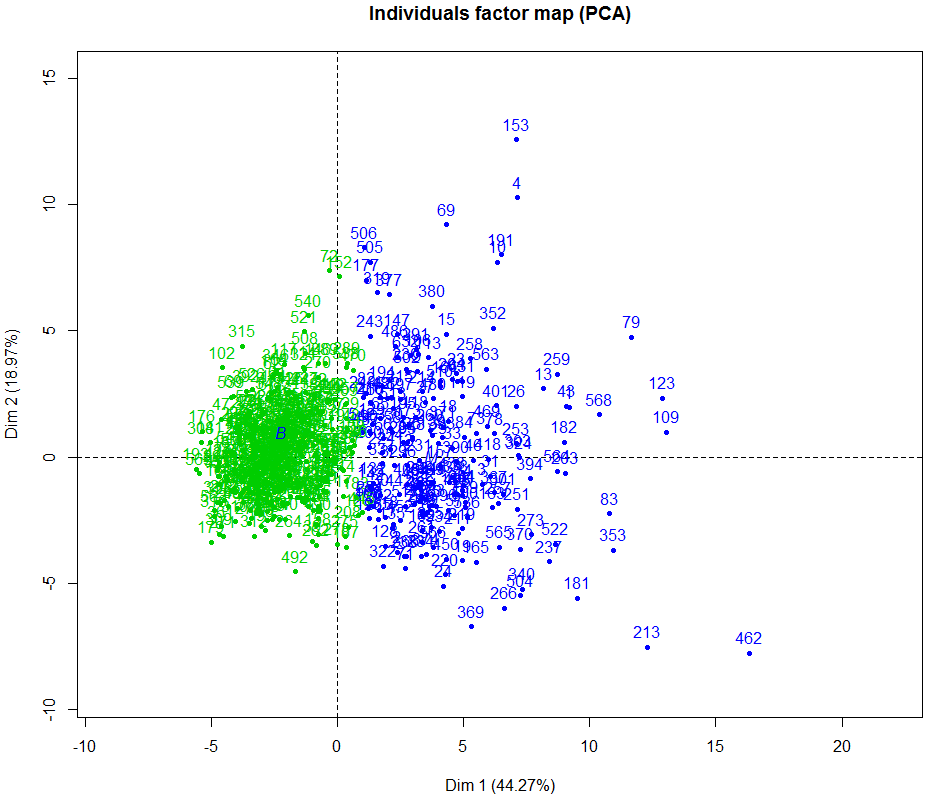
The two clusters formed using k-means clustering () almost perfectly coincide with cancer status of the cells.

PCA in the FactoMineR package is probably the most comprehensive that I have found in R libraries. It allows for specification of supplementary variables that will not be used in computing the principal components, but rather will be used in displaying the results. In the function below I have specified that diagnosis (malignant or benign) is a qualitative (i.e. nominal/categorical) supplementary variable. As we can see from above this information is the first variable in the data frame BreastDiag, thus quali.sup = 1.

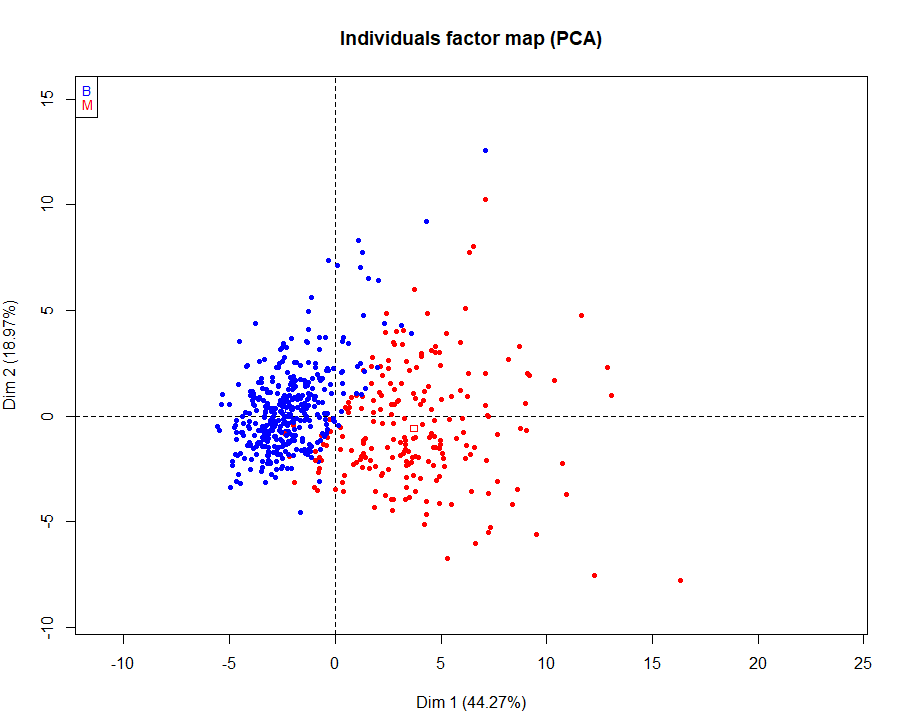
> library(FactoMineR)

> breast.PCA = PCA(BreastDiag,quali.sup=1) 🡨 performs PCA using PCA function   
 from the FactoMineR package.

> plot(breast.PCA,col.ind=as.numeric(breast.kmeans$cluster)+2)



> plot(breast.PCA,choix="ind",habillage=1,col.hab=c("blue","red"),label="none")



**PCA Revisited - Contribution and Quality of Representation for Variables and Individuals**

In PCA analysis here are some questions we might wish to answer based on our results:

* How much does each variable **contribute** to a principal component of interest?
* How much variation of a given variable do the first PC’s account for?   
  The measures the quality of representation of this variable in the lower dimensional space.
* How much does a given individual contribute to the PCA? Outliers or extreme cases can have undue influence on the results from a PCA.
* How well represented is a given individual in the -dimensional representation?

We will examine the answer to these questions for the PCA of the breast cancer data set. Packages FactoMineR and factoextra have functions which allow is to obtain the metrics needed used to answer questions for variables and individuals.

**Variables – importance and quality of representation**For variables we use the function get\_pca\_var(*PCA analysis object*). For the breast cancer data set we have the following.

> var.results = get\_pca\_var(breast.PCA)

> var.results

Principal Component Analysis Results for variables

===================================================

Name Description

1 "$coord" "Coordinates for the variables"

2 "$cor" "Correlations between variables and dimensions"

3 "$cos2" "Cos2 for the variables"

4 "$contrib" "contributions of the variables"  
  
> var.results$coord

Dim.1 Dim.2 Dim.3 Dim.4 Dim.5

Radius 0.79776675 -0.55790267 -0.014321182 -0.058276998 -0.048518775

Texture 0.37801323 -0.14243819 0.108358294 0.848703801 0.063519440

Perimeter 0.82923555 -0.51334871 -0.015635546 -0.059085011 -0.047990153

Area 0.80539280 -0.55126955 0.048177170 -0.075200175 -0.013265627

Smoothness 0.51965303 0.44400165 -0.175072188 -0.224307700 0.468784268

Compactness 0.87205011 0.36236113 -0.124375651 -0.044746177 -0.015028239

… … … … … …

wsmoo 0.46630955 0.41105891 -0.436115698 -0.024842887 0.416584530

wcomp 0.76567217 0.34256392 -0.396294196 0.128531255 -0.156400009

wconc 0.83371903 0.23370868 -0.290506982 0.104075362 -0.242063518

wconpts 0.91432733 -0.01969892 -0.285952302 -0.008453959 -0.055639634

wsym 0.44791263 0.33848486 -0.455445686 0.051017498 0.314020423

wfd 0.48027261 0.65686526 -0.390780903 0.108441378 -0.121242324

These are scores on the first principal components where the variables are viewed as the observations and the individuals are viewed as the variables.

The correlation between variable and the scores for the *k*-th principal component can be used to measure the importance of that variable for that PC. Below is a table of the correlations between in of the variables and the first principal components.

> var.results$cor

Dim.1 Dim.2 Dim.3 Dim.4 Dim.5

Radius 0.79776675 -0.55790267 -0.014321182 -0.058276998 -0.048518775

Texture 0.37801323 -0.14243819 0.108358294 0.848703801 0.063519440

Perimeter 0.82923555 -0.51334871 -0.015635546 -0.059085011 -0.047990153

Area 0.80539280 -0.55126955 0.048177170 -0.075200175 -0.013265627

Smoothness 0.51965303 0.44400165 -0.175072188 -0.224307700 0.468784268

Compactness 0.87205011 0.36236113 -0.124375651 -0.044746177 -0.015028239

Concavity 0.94171317 0.14353386 0.004589225 -0.026912451 -0.110908536

ConcavePts 0.95065387 -0.08294330 -0.042912871 -0.091950691 0.056318830

… … … … … …

wcomp 0.76567217 0.34256392 -0.396294196 0.128531255 -0.156400009

wconc 0.83371903 0.23370868 -0.290506982 0.104075362 -0.242063518

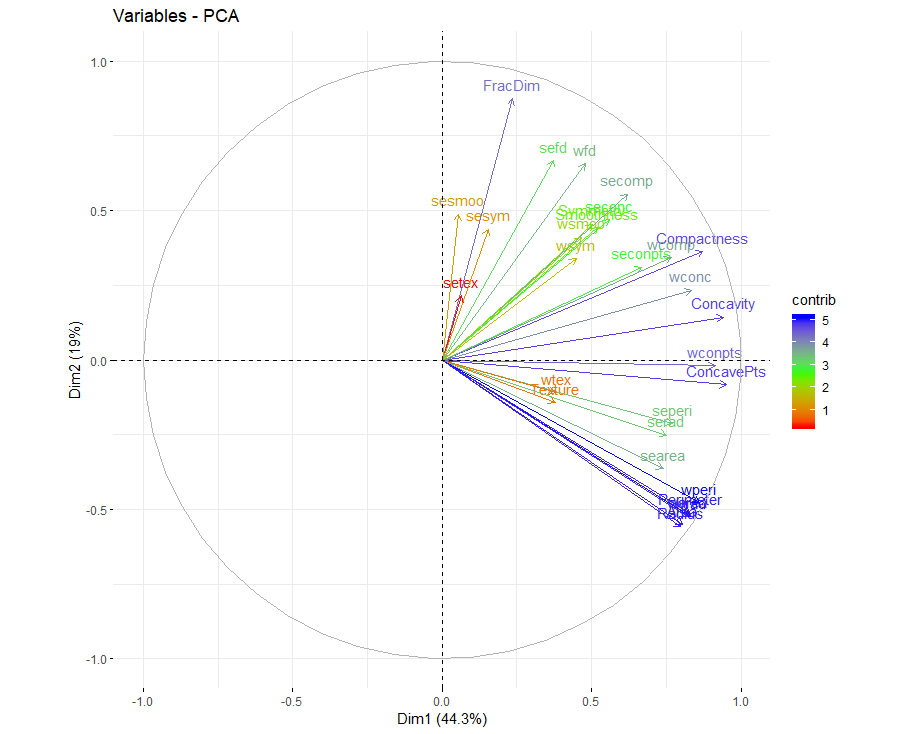
wconpts 0.91432733 -0.01969892 -0.285952302 -0.008453959 -0.055639634

wsym 0.44791263 0.33848486 -0.455445686 0.051017498 0.314020423

wfd 0.48027261 0.65686526 -0.390780903 0.108441378 -0.121242324

We can see that for the first principal component Compactness, Concavity, ConcavePts, and wconpts have a very high correlation with the PC1 scores, which speaks to the importance of these variables in determining the PC1 scores.

> fviz\_pca\_var(breast.PCA, col.var = "contrib",gradient.cols = c("red", "green", "blue))



Another measure used to measure the importance of a variable in determining the values of is to consider the cosine-squared measure which is simply the square of the loading on for the variable divided but the sum of the squared loadings of all variables on . This is called the **contribution** of to .

The contribution can be expressed as a percentage. If all variables contribute equally to a component, then we would expect the contribution to be for each. Here , so if all contribute equally to a given component, we expect each to have a contribution of Below are the contributions of all variables for each of the first principal components.

> var.results$contrib

Dim.1 Dim.2 Dim.3 Dim.4 Dim.5

Radius 4.79182799 5.468915807 0.007278210 1.714702e-01 0.142780853

Texture 1.07587881 0.356481698 0.416669002 3.636693e+01 0.244716717

Perimeter 5.17732197 4.630301829 0.008675469 1.762581e-01 0.139686545

Area 4.88387836 5.339644650 0.082366279 2.855170e-01 0.010673476

Smoothness 2.03318209 3.463805721 1.087680124 2.540287e+00 13.328963320

Compactness 5.72574806 2.307106121 0.548956087 1.010895e-01 0.013698294

Concavity 6.67708087 0.361987088 0.000747387 3.656797e-02 0.746071176

ConcavePts 6.80446833 0.120877909 0.065349461 4.268786e-01 0.192378954

Symmetry 1.90901086 3.623265438 0.161925247 4.505763e-01 9.360015755

FracDim 0.41426404 13.437757622 0.050958953 2.360674e-01 0.197352378

serad 4.24272561 1.114125676 7.208225536 9.592487e-01 2.385680908

setex 0.03037362 0.809634314 14.035038303 1.294960e+01 3.672991628

seperi 4.46586429 0.800259675 7.109975188 7.919650e-01 1.463863342

searea 4.11560890 2.319304458 4.665882031 1.170833e+00 1.627523570

sesmoo 0.02111631 4.179181013 9.538151522 1.994889e-01 5.385447803

secomp 2.90339282 5.415668834 2.395676105 7.545659e-02 7.838216812

seconc 2.35898235 3.889071235 3.113945256 1.734173e-04 12.530332072

seconpts 3.36419415 1.698370898 5.047102257 5.485970e-01 3.823905492

sesym 0.18061158 3.380008708 8.328089385 1.942460e-01 6.394261226

sefd 1.05202607 7.845154337 4.473384240 2.342354e-02 6.932554069

wrad 5.19824652 4.834122475 0.225691410 2.376913e-02 0.001941805

wtex 1.09138400 0.206727521 0.178910581 4.004458e+01 0.862732602

wperi 5.59983385 3.995138596 0.235676347 1.905171e-02 0.005556437

warea 5.05667565 4.811523757 0.014166518 6.705380e-02 0.075026157

wsmoo 1.63718579 2.968878960 6.749479988 3.116007e-02 10.525835817

wcomp 4.41402789 2.061899941 5.573170087 8.340879e-01 1.483624039

wconc 5.23345841 0.959696770 2.994884133 5.468777e-01 3.553931037

wconpts 6.29437706 0.006818193 2.901710407 3.608400e-03 0.187766818

wsym 1.51055300 2.013088461 7.361054968 1.314113e-01 5.980893978

wfd 1.73670076 7.581182296 5.419179518 5.937237e-01 0.891576921

The ***quality of representation*** of variable on a given principal component on the first principal components is given by the cosine-squared measure (or ). When considering the **quality of representation** of the variables, we again need to think of our PC analysis as being conducted on the transpose of the data matrix, i.e. the rows (observations) are the variables and the columns (variables) are the individuals.

**Idea:** The cosine of the angle between two vectors is given by:

If **u** and **v** are close together then the angle between them will be near 0, and the will be near 1. In PCA when using this as a measure of quality of representation, we are considering the angle between the data in the original full dimension and the reduced dimensional representation from PCA. Variables that are well represented in the lower dimensional space with have and hence values near 1. An analogous measure can be used when discussing quality of representation for individuals as well (which we will consider next for these data).

Below are the values for the breast cancer PCA with rows as variables.

> var.results$cos2

[1,] 0.94768719

[2,] 0.16318264

[3,] 0.95115849

[4,] 0.95255568

[5,] 0.46717674

[6,] 0.89177699

[7,] 0.90742565

[8,] 0.91062238

[9,] 0.45976022

[10,] 0.81981136

[11,] 0.62691101

[12,] 0.05011326

[13,] 0.63868419

[14,] 0.67861887

[15,] 0.24065660

[16,] 0.69384216

[17,] 0.53465162

[18,] 0.54347938

[19,] 0.21635640

[20,] 0.58622153

[21,] 0.96553776

[22,] 0.15671894

[23,] 0.97112547

[24,] 0.94544870

[25,] 0.38641402

[26,] 0.70360390

[27,] 0.74970716

[28,] 0.83638251

[29,] 0.31519773

[30,] 0.66213375

Dim.1 Dim.2 Dim.3 Dim.4 Dim.5

Radius 0.636431794 0.3112553921 2.050963e-04 3.396209e-03 2.354072e-03

Texture 0.142894003 0.0202886376 1.174152e-02 7.202981e-01 4.034719e-03

Perimeter 0.687631593 0.2635268968 2.444703e-04 3.491038e-03 2.303055e-03

Area 0.648657563 0.3038981121 2.321040e-03 5.655066e-03 1.759769e-04

Smoothness 0.270039269 0.1971374667 3.065027e-02 5.031394e-02 2.197587e-01

Compactness 0.760471394 0.1313055907 1.546930e-02 2.002220e-03 2.258480e-04

Concavity 0.886823686 0.0206019688 2.106098e-05 7.242800e-04 1.230070e-02

ConcavePts 0.903742788 0.0068795905 1.841514e-03 8.454930e-03 3.171811e-03

Symmetry 0.253547334 0.2062128847 4.562971e-03 8.924298e-03 1.543214e-01

FracDim 0.055020924 0.7647904383 1.435997e-03 4.675646e-03 3.253809e-03

serad 0.563502171 0.0634088430 2.031241e-01 1.899927e-02 3.933345e-02

setex 0.004034105 0.0460791599 3.955002e-01 2.564850e-01 6.055773e-02

seperi 0.593138575 0.0455456160 2.003555e-01 1.568598e-02 2.413516e-02

searea 0.546619028 0.1319998412 1.314822e-01 2.318999e-02 2.683348e-02

sesmoo 0.002804585 0.2378520114 2.687802e-01 3.951158e-03 8.879152e-02

secomp 0.385617244 0.3082249180 6.750893e-02 1.494524e-03 1.292311e-01

seconc 0.313310781 0.2213408352 8.774939e-02 3.434773e-06 2.065914e-01

seconpts 0.446819069 0.0966603104 1.422248e-01 1.086573e-02 6.304590e-02

sesym 0.023988122 0.1923682816 2.346813e-01 3.847315e-03 1.054241e-01

sefd 0.139725975 0.4464955533 1.260577e-01 4.639360e-04 1.142991e-01

wrad 0.690410709 0.2751270525 6.359869e-03 4.707810e-04 3.201514e-05

wtex 0.144953341 0.0117655963 5.041609e-03 7.931392e-01 1.422414e-02

wperi 0.743747963 0.2273775048 6.641239e-03 3.773459e-04 9.161067e-05

warea 0.671607821 0.2738408793 3.992052e-04 1.328095e-03 1.236979e-03

wsmoo 0.217444594 0.1689694296 1.901969e-01 6.171690e-04 1.735427e-01

wcomp 0.586253867 0.1173500374 1.570491e-01 1.652028e-02 2.446096e-02

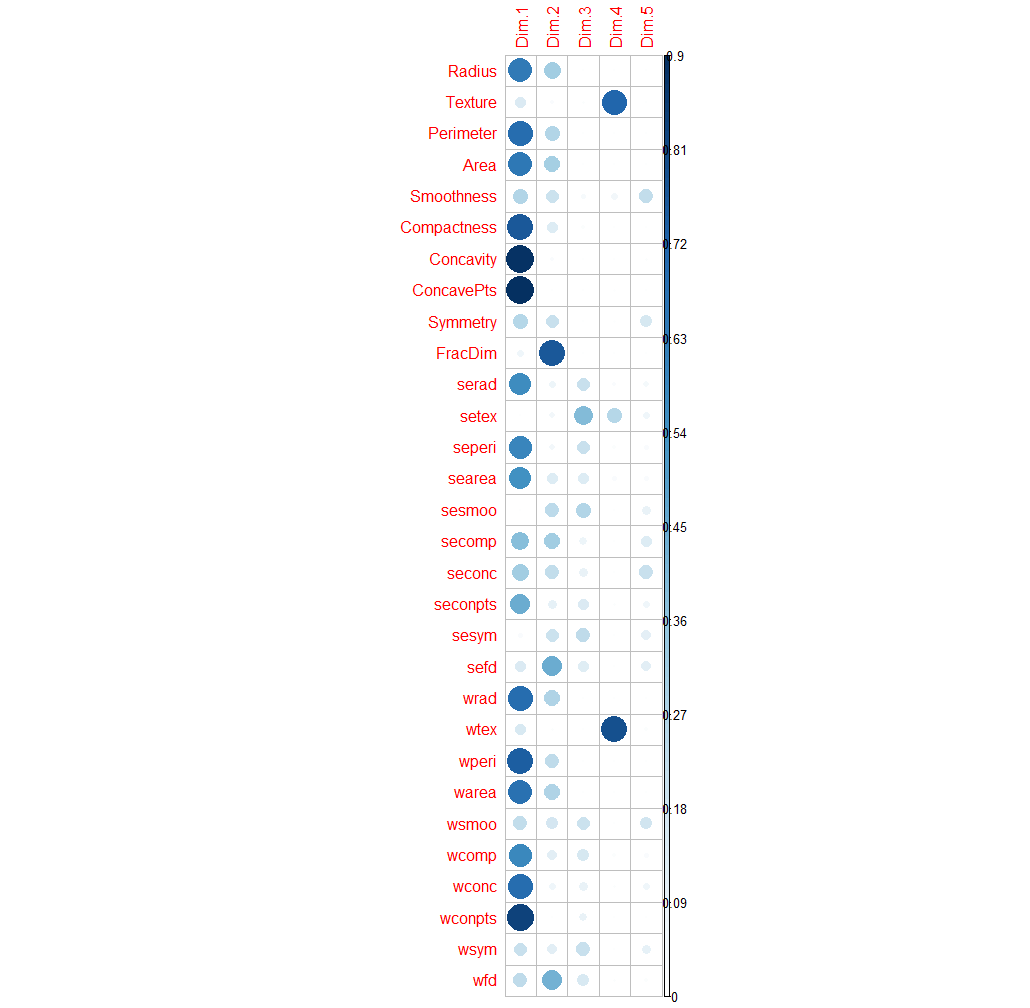
wconc 0.695087414 0.0546197464 8.439431e-02 1.083168e-02 5.859475e-02

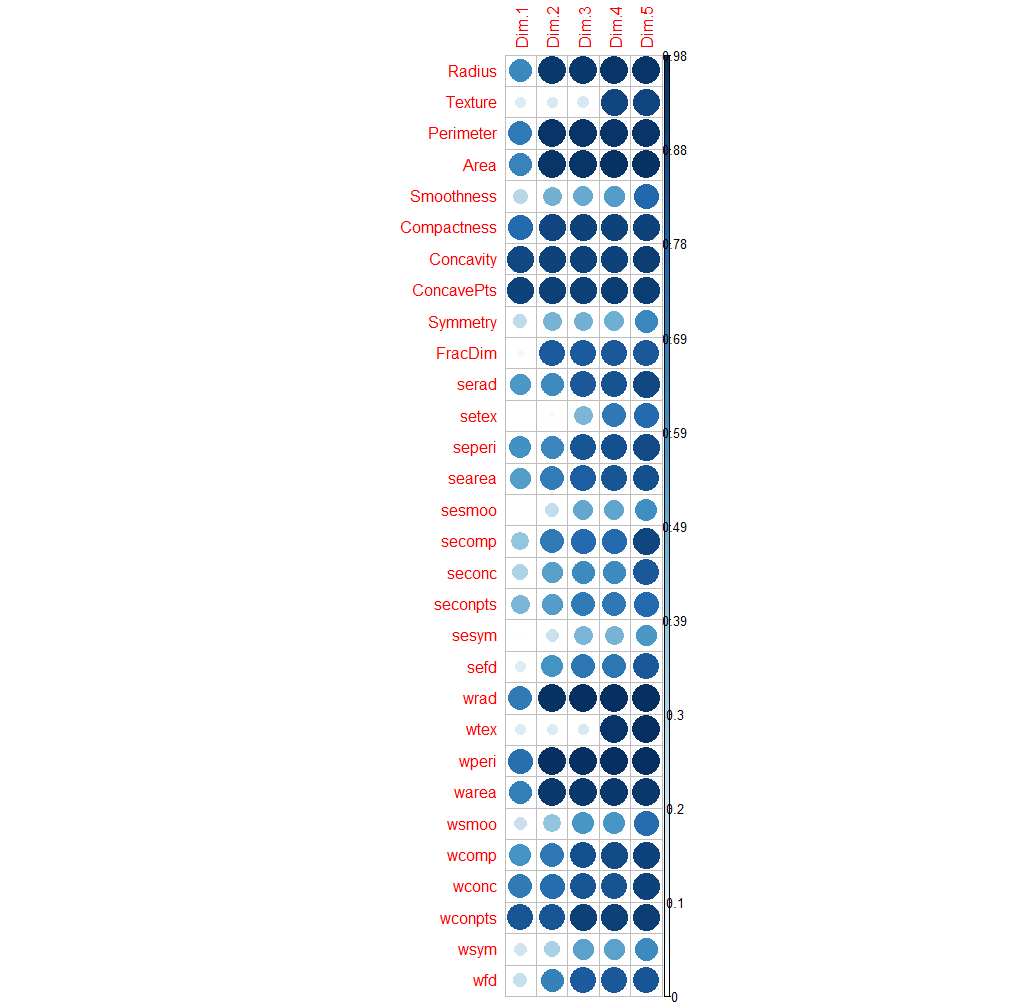
wconpts 0.835994467 0.0003880475 8.176872e-02 7.146943e-05 3.095769e-03

wsym 0.200625723 0.1145720030 2.074308e-01 2.602785e-03 9.860883e-02

wfd 0.230661782 0.4314719683 1.527097e-01 1.175953e-02 1.469970e-02

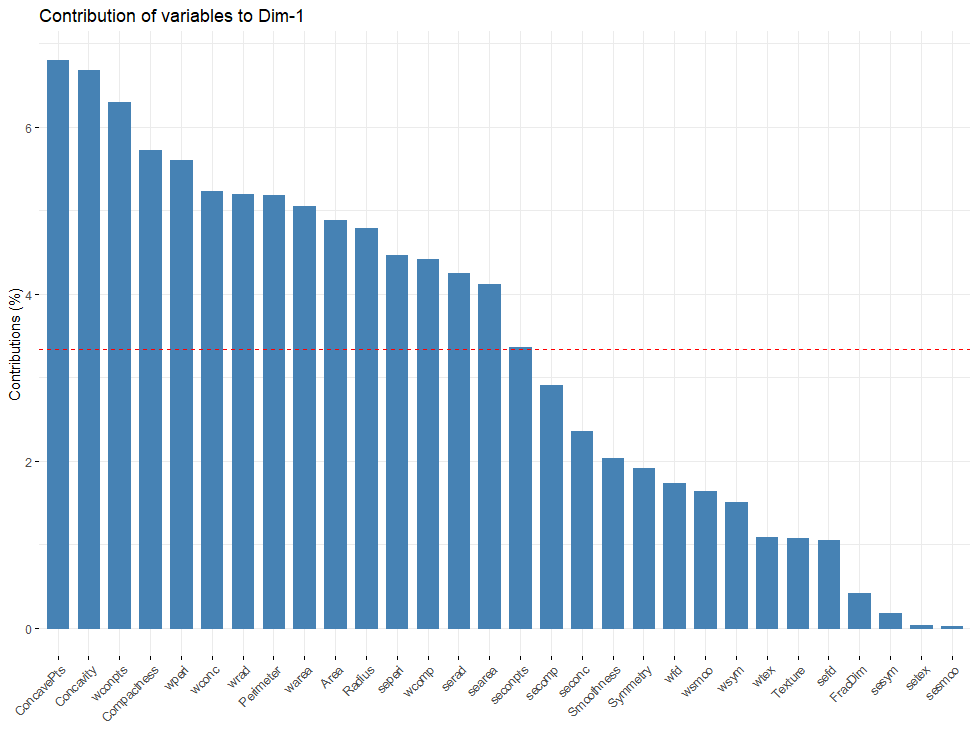
We can visualize the cosine squared values on each dimension using corrplot.

> corrplot(var.results$cos2,is.corr=F)  


> corrplot(t(apply(var.results$cos2,1,cumsum)),is.corr=F)  


This plot shows the cumulative sum of the cosine square values as we increase dimensions of our PCA. We can see several variables exceed 90% using only .

> fviz\_contrib(breast.PCA,choice=”var”)



For more information on the FactoMineR, FactoInvestigate, and factoextra see the following links:

* FactoMineR - <http://factominer.free.fr/>
* Contribution and quality of representation - <http://www.sthda.com/english/wiki/print.php?id=204>
* FactoInvestigate - <http://factominer.free.fr/reporting/index.html> 🡨 check this out, PCA for dummies
* YouTube – Francois Husson – he is the developer of all things Facto and has GREAT videos!

**Individuals – contribution and quality of representation**

We now consider individuals in terms of contribution to the PCA solution and the quality of representation of the individuals in a lower dimensional space. The measures are essential the same as for variables, but here we are returning to the usual format of our data matrix where rows are individuals and columns are the variables measured on the individuals.

> ind.results = get\_pca\_ind(breast.PCA)  
> ind.results

Principal Component Analysis Results for individuals

===================================================

Name Description

1 "$coord" "Coordinates for the individuals"

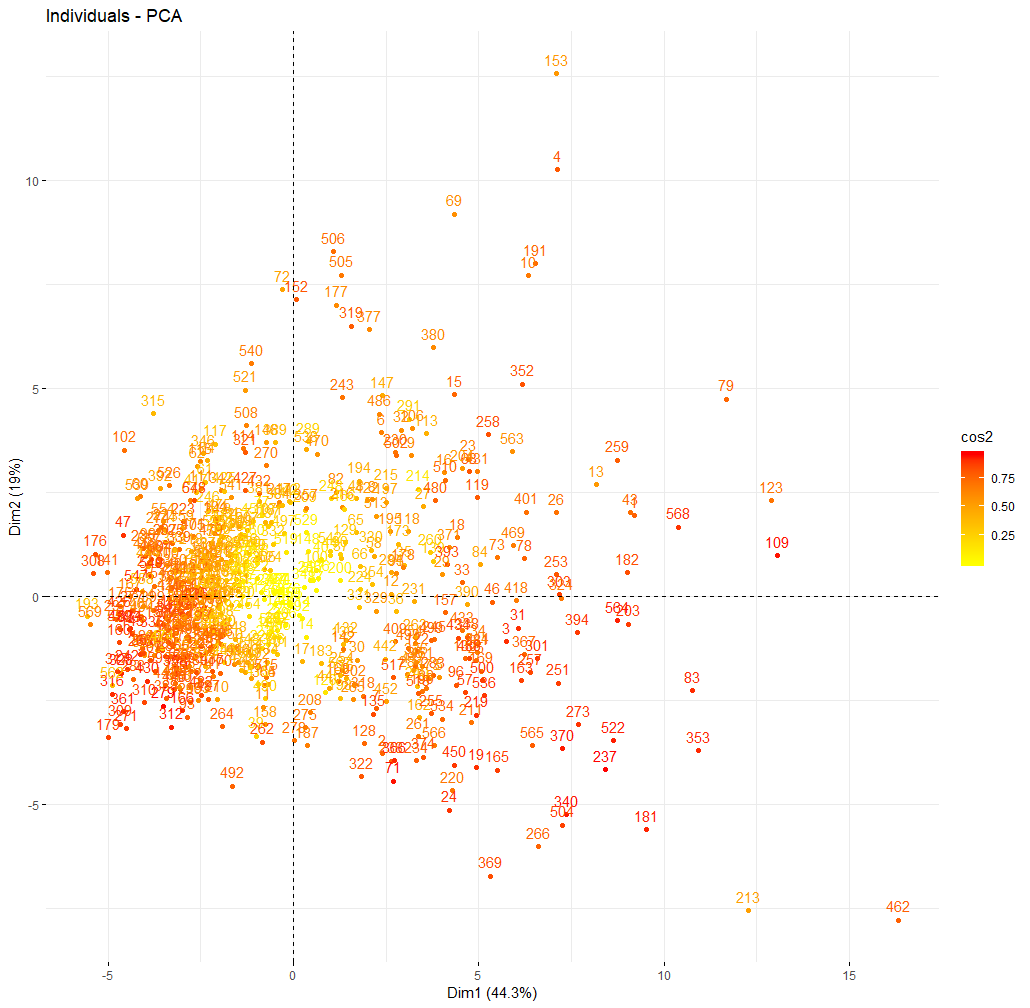
2 "$cos2" "Cos2 for the individuals"

3 "$contrib" "contributions of the individuals"

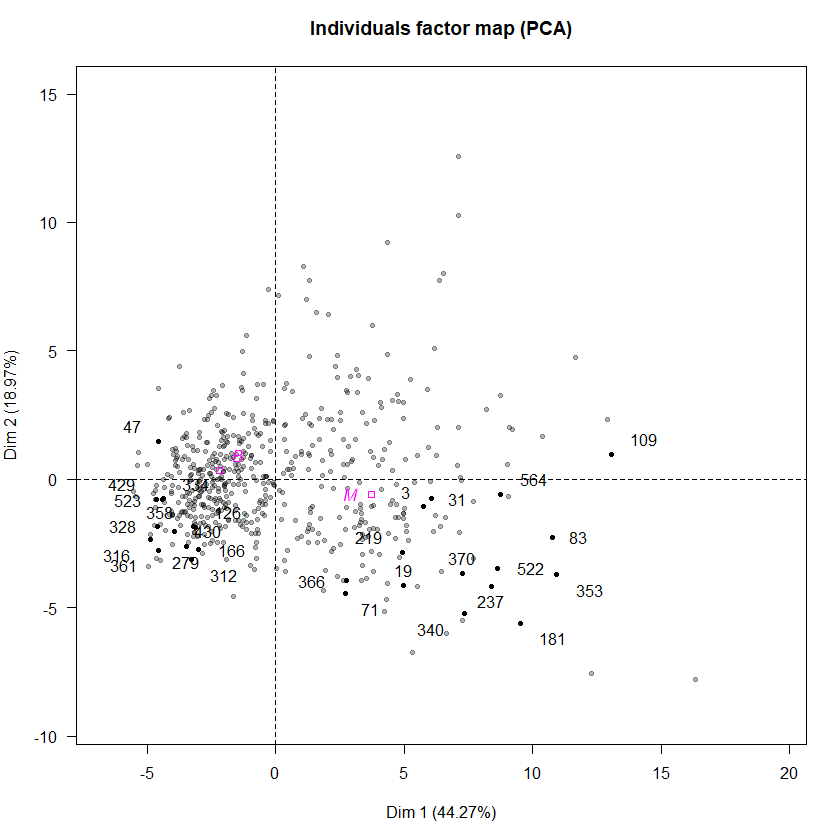
fviz\_pca\_ind(breast.PCA, col.ind = "cos2",

gradient.cols = c("yellow","orange","red"),

repel = FALSE # To avoid text overlapping set to TRUE (slow if many points)) )



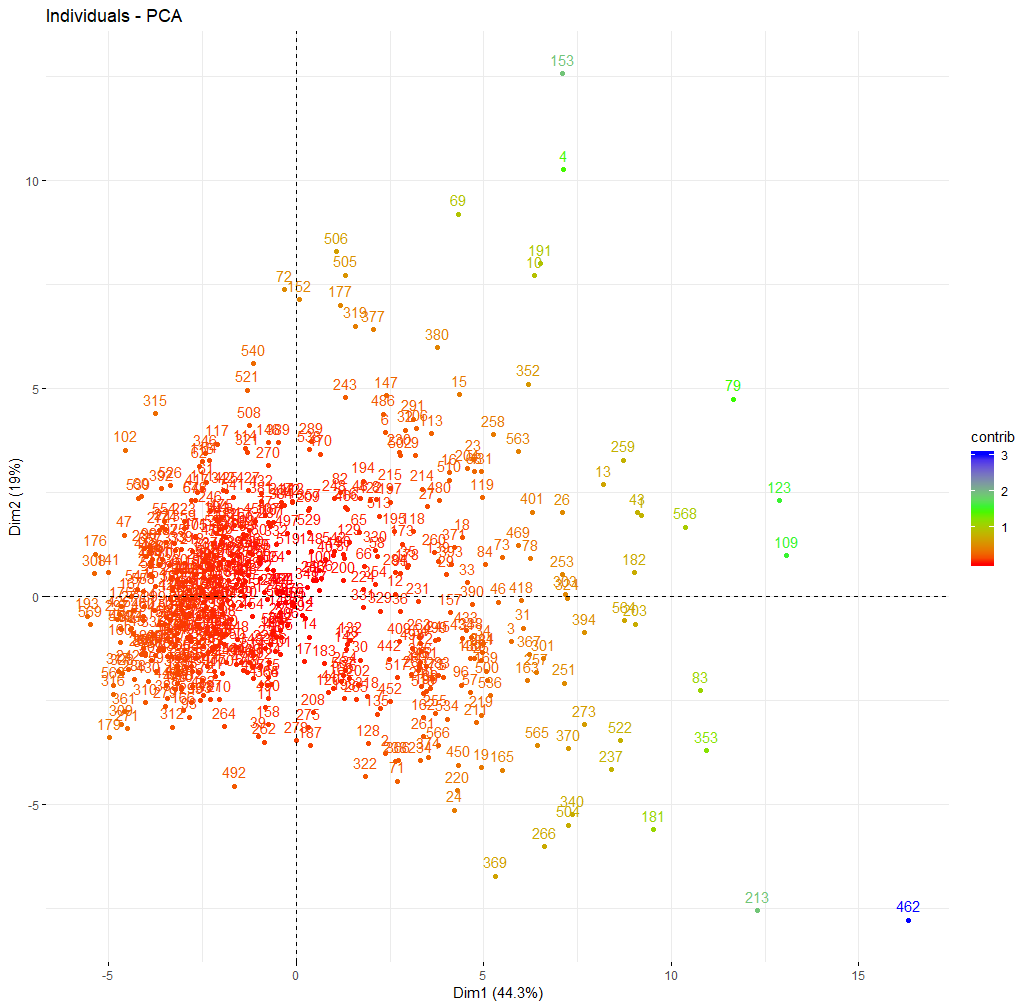
plot(breast.PCA,choix="ind",select="cos2 0.9")



fviz\_pca\_ind(breast.PCA, col.ind = "contrib",

gradient.cols = c("red","green","blue"),

repel = FALSE # Avoid text overlapping (slow if many points))

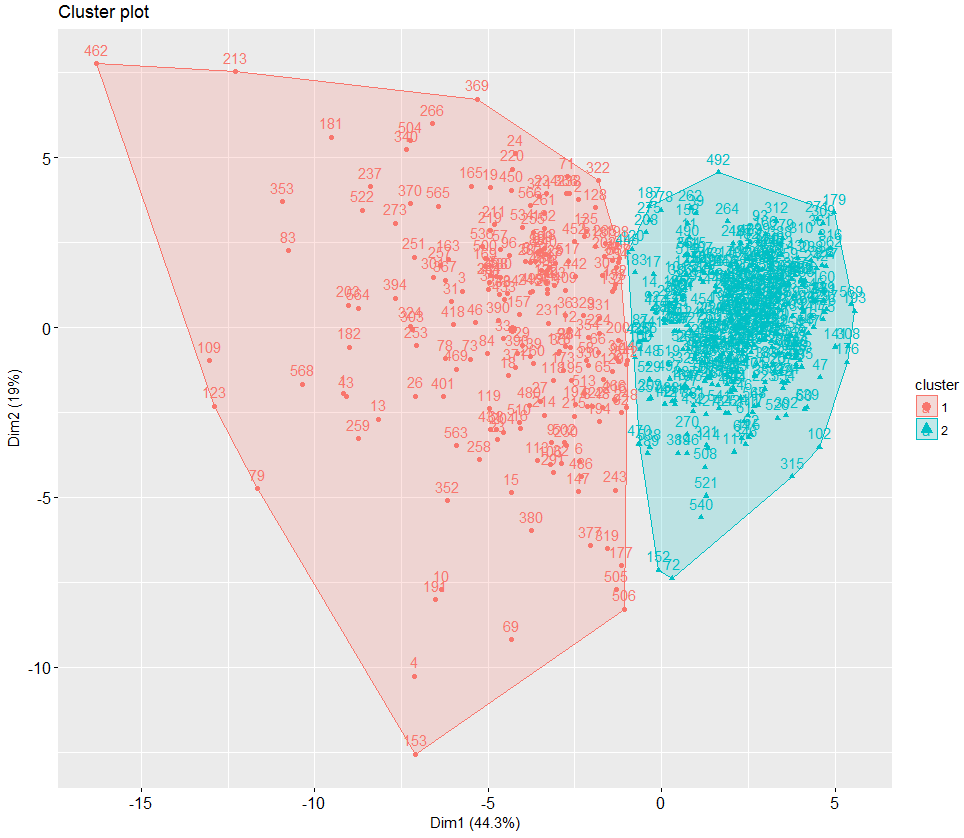
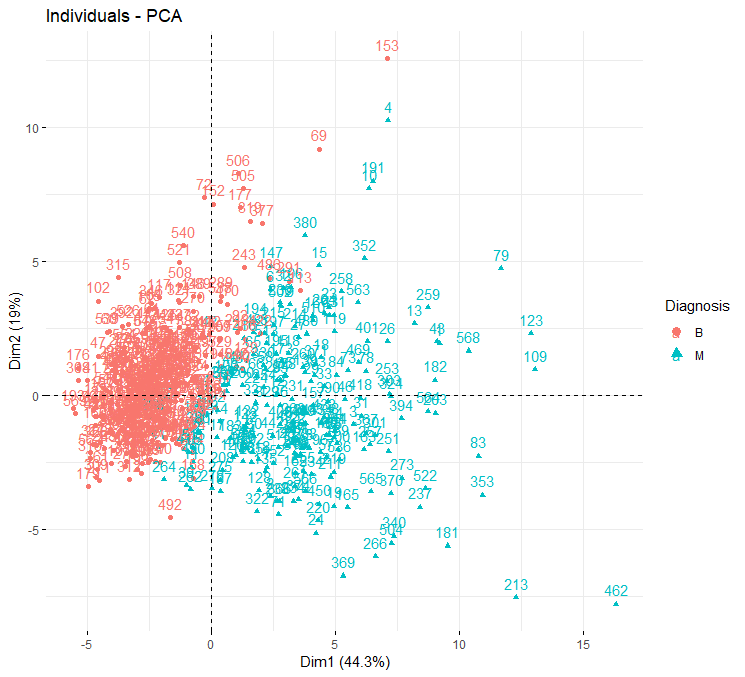


The function fviz\_cluster() in the factoextra library provides a very nice visualization of the clusters returned from partitioning methods such as k-means and PAM.

> library(factoextra)

> breast.kmeans = kmeans(breast.mat,2)

> fviz\_cluster(breast.kmeans,breast.mat)  
> fviz\_pca\_ind(breast.PCA,habillage=1)

While neither technique, PCA or clustering, is attempting to classify benign or malignant tumors they do separate the two types of cells very well.

Example 5.6: Sports Difficulty Data

We will examine different methods of hierarchical clustering of these data before considering k-means.

> names(SportsDiff)

[1] "Endurance" "Strength" "Power" "Speed"

[5] "Agility" "Flexibility" "Nerves" "Durability"

[9] "Hand.Eye" "AnalyticAptitude" "Total\_Score" "Rank"

> head(SportsDiff)

Endurance Strength Power Speed Agility Flexibility Nerves Durability Hand.Eye

Boxing 8.63 8.13 8.63 6.38 6.25 4.38 8.88 8.50 7.00

Ice Hockey 7.25 7.13 7.88 7.75 7.63 4.88 6.00 8.25 7.50

Football 5.38 8.63 8.13 7.13 6.38 4.38 7.25 8.50 5.50

Basketball 7.38 6.25 6.50 7.25 8.13 5.63 4.13 7.75 7.50

Wrestling 6.63 8.38 7.13 5.13 6.38 7.50 5.00 6.75 4.25

Martial Arts 5.00 5.88 7.75 6.38 6.00 7.00 6.63 5.88 6.00

AnalyticAptitude Total\_Score Rank

Boxing 5.63 72.375 1

Ice Hockey 7.50 71.750 2

Football 7.13 68.375 3

Basketball 7.38 67.875 4

Wrestling 6.38 63.500 5

Martial Arts 6.88 63.375 6

Agglomerative Hierarchical Clustering

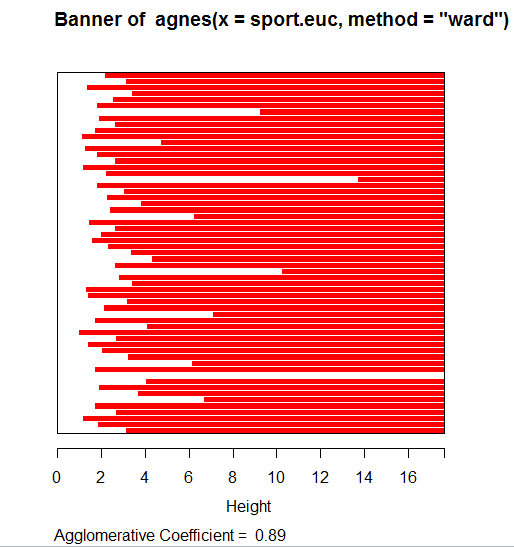
> sport.mat = scale(SportsDiff[,-c(11,12)])

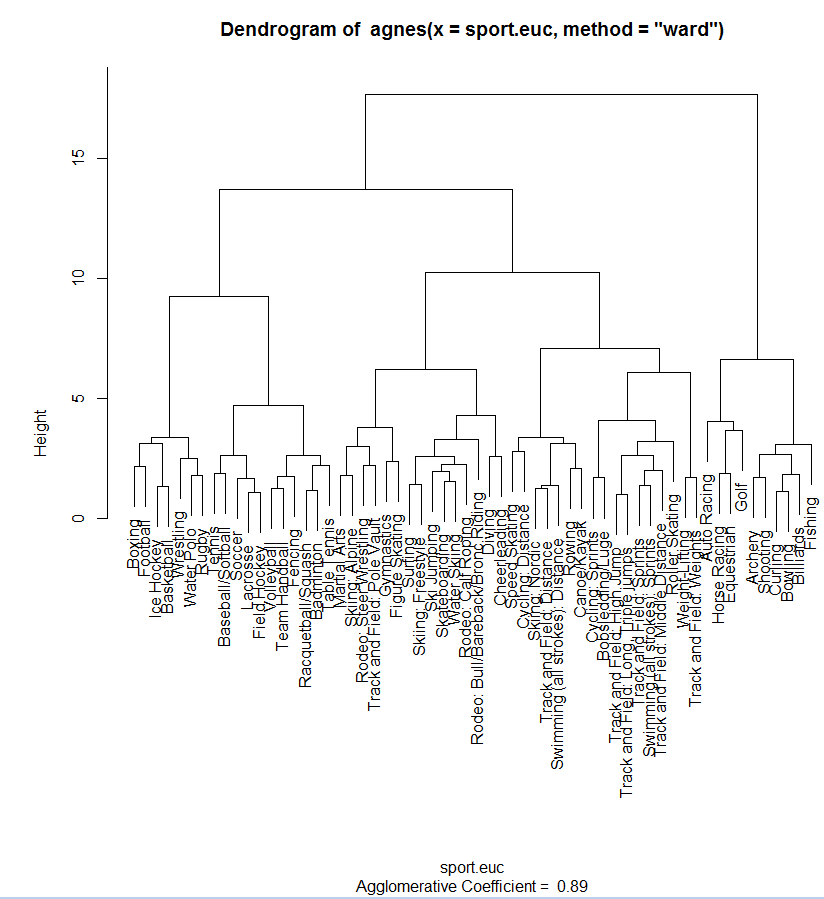
> sport.euc = dist(sport.mat)

> sport.man = dist(sport.mat,method=”manhattan”)

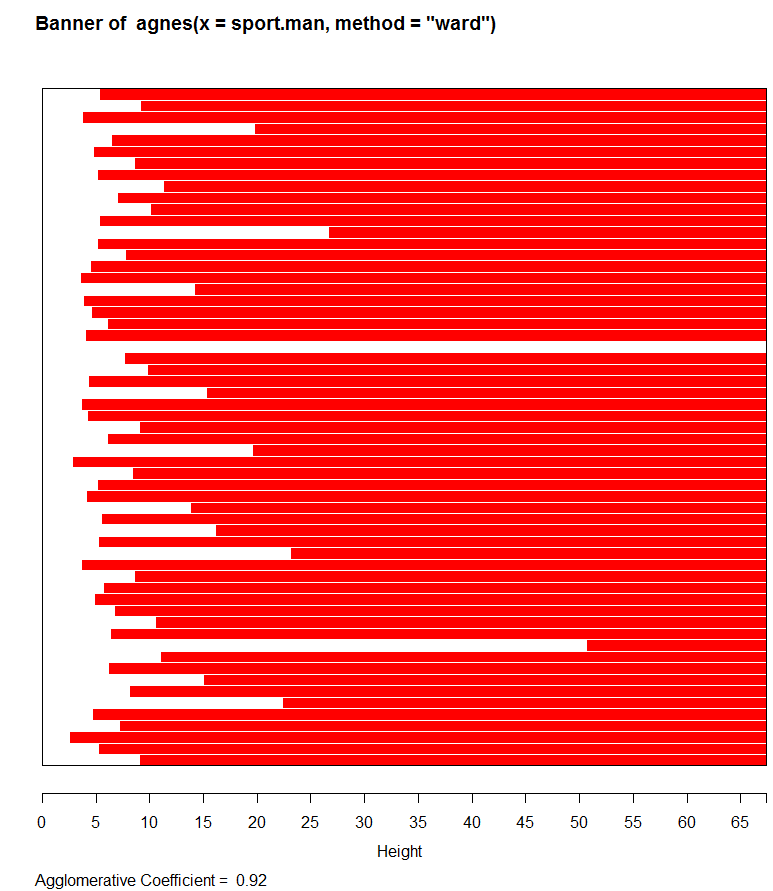
> sp1 = agnes(sport.euc,method="ward")

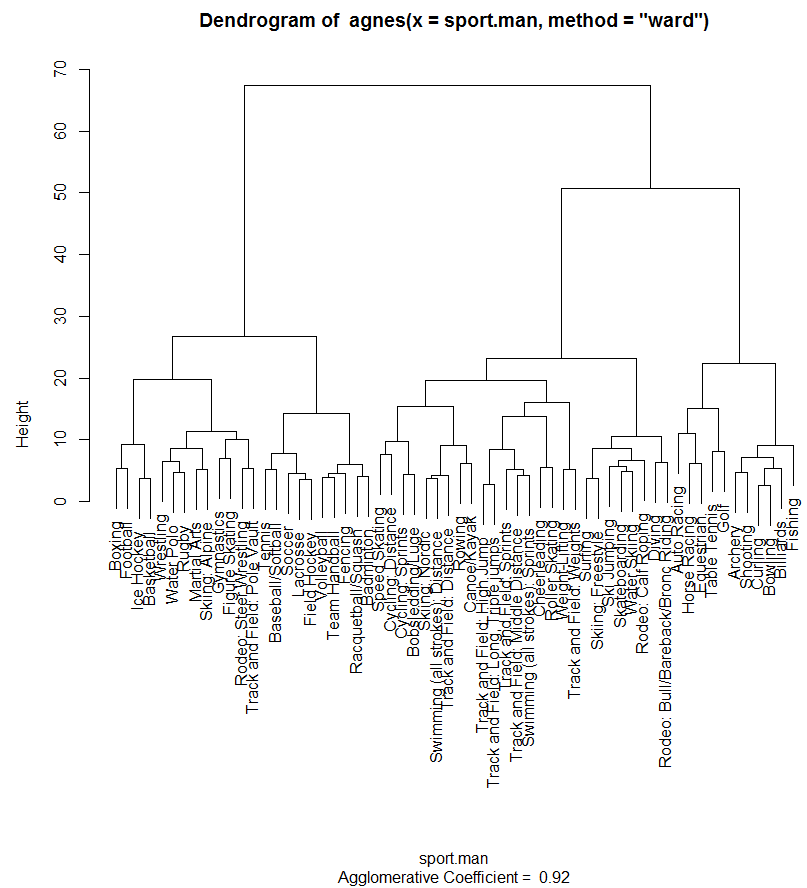
> plot(sp1)





> sp2 = agnes(sport.man,method=”ward”)

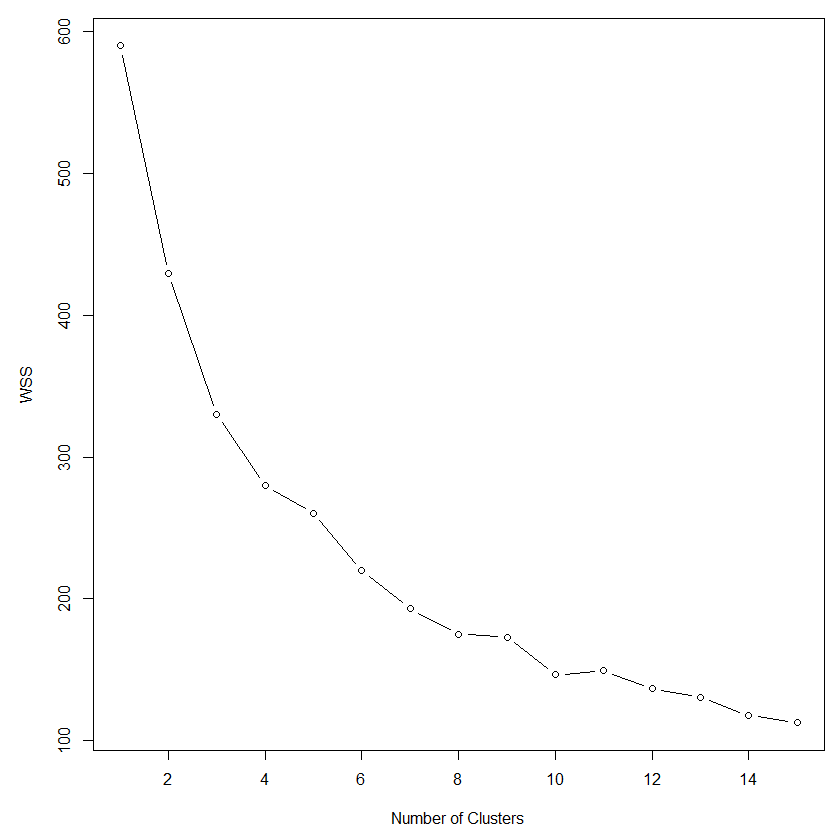
> plot(sp2)  




K-Means

> wss = rep(0,15)

> for (i in 1:15) wss[i]=sum(kmeans(sport.mat,centers=i)$withinss)  
> plot(1:15,wss,xlab=”Number of Clusters”,ylab=”WSS”,type=”b”)

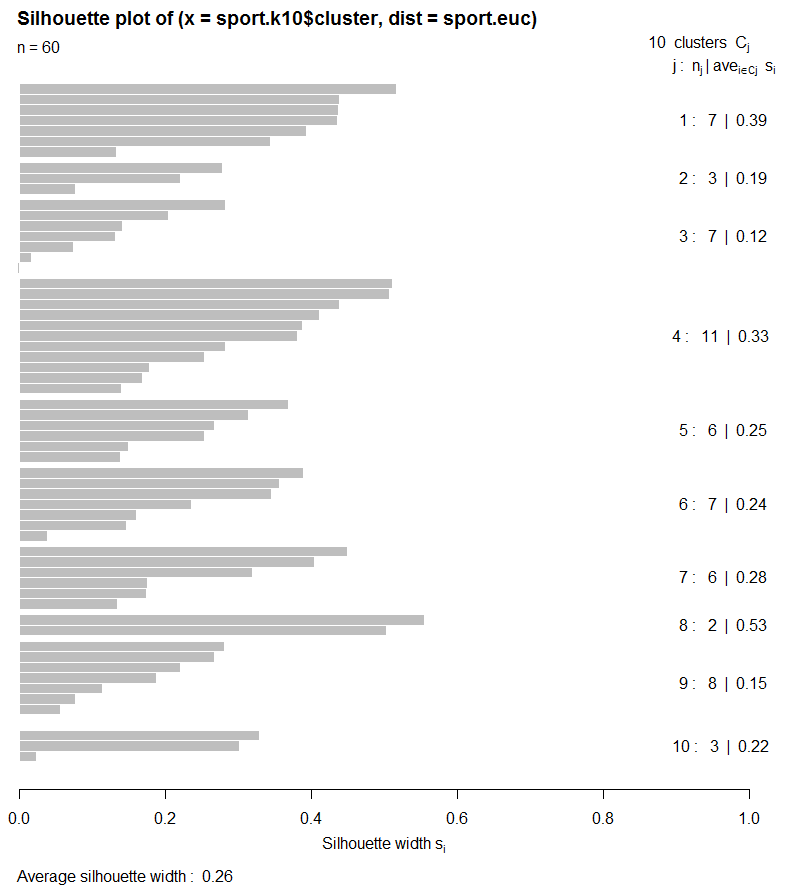


We can use this plot to help choose the number of clusters to use in our k-mean clustering solution.

> sport.k10 = kmeans(sport.mat,10)

> si = silhouette(sport.k10$cluster,sport.euc)

> plot(si)



> clust.grps(SportsDiff,sport.k10$cluster)

Cluster 1

=======================================================================

Golf Archery Curling Bowling Shooting Billiards Fishing

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

2.180000 2.698571 2.788571 1.074286 1.591429 2.450000

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

1.931429 1.538571 5.234286 4.574286 26.035714 56.571429

Cluster 2

=======================================================================

Speed Skating Cycling: Sprints Bobsledding/Luge

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

5.126667 6.293333 7.253333 7.710000 4.043333 3.460000

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

5.666667 4.210000 3.546667 4.083333 51.375000 24.000000

Cluster 3

=======================================================================

Martial Arts Gymnastics Skiing: Alpine Rodeo: Steer Wrestling Track and Field: Pole Vault Figure Skating Skiing: Freestyle

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

4.771429 5.931429 6.717143 5.575714 6.040000 7.108571

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

6.932857 5.574286 4.752857 4.645714 58.017857 14.285714

Cluster 4

=======================================================================

Tennis Baseball/Softball Soccer Lacrosse Field Hockey Volleyball Racquetball/Squash Fencing Team Handball Badminton Table Tennis

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

5.684545 4.274545 5.537273 5.853636 6.797273 5.013636

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

3.241818 4.378182 7.707273 6.468182 54.931818 20.545455

Cluster 5

=======================================================================

Track and Field: High Jump Track and Field: Long, Triple jumps Track and Field: Sprints Track and Field: Middle Distance Swimming (all strokes): Sprints Roller Skating

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

4.230000 5.086667 6.126667 7.253333 4.481667 5.231667

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

2.563333 3.710000 2.981667 2.961667 44.604167 40.833333

Cluster 6

=======================================================================

Boxing Ice Hockey Football Basketball Wrestling Water Polo Rugby

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

7.128571 7.450000 7.361429 6.414286 6.735714 5.128571

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

6.001429 7.715714 6.054286 6.468571 66.428571 5.571429

Cluster 7

=======================================================================

Cycling: Distance Skiing: Nordic Swimming (all strokes): Distance Rowing Track and Field: Distance Canoe/Kayak

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

8.731667 5.938333 5.295000 4.876667 3.313333 4.065000

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

3.106667 5.065000 2.900000 3.961667 47.229167 35.666667

Cluster 8

=======================================================================

Weight-Lifting Track and Field: Weights

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

3.6900 8.5650 9.4400 2.8150 2.8150 3.1900

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

3.1250 4.1900 3.1250 2.6300 43.5625 46.0000

Cluster 9

=======================================================================

Surfing Ski Jumping Diving Skateboarding Rodeo: Calf Roping Rodeo: Bull/Bareback/Bronc Riding Water Skiing Cheerleading

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

3.72250 4.72125 4.39250 3.40750 4.73750 5.56375

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

7.00250 4.94000 4.16000 3.36125 45.98438 38.00000

Cluster 10

=======================================================================

Auto Racing Horse Racing Equestrian

Variable means in this cluster are:

-----------------------------------------------------------------------

Endurance Strength Power Speed Agility Flexibility

4.420000 3.543333 2.420000 1.420000 2.710000 2.793333

Nerves Durability Hand.Eye AnalyticAptitude Total\_Score Rank

7.960000 3.876667 4.920000 6.376667 40.416667 45.333333

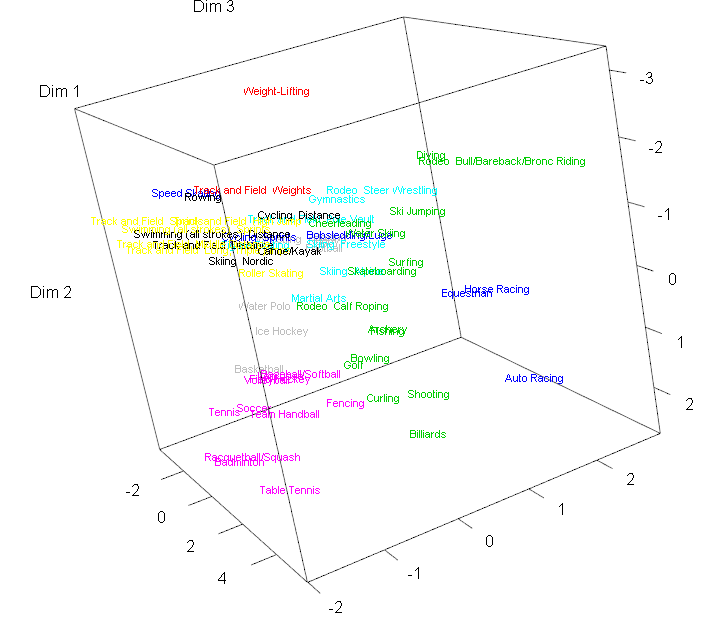
To potentially better understand clusters we can look at MDS.

> sport.mds = cmdscale(sport.euc,k=3)

> plot3d(sport.mds,type="n",xlab="Dim 1",ylab="Dim 2",zlab="Dim 3")

> text3d(sport.mds,texts=row.names(SportsDiff),

col=sport.k10$cluster+2,cex=.7)



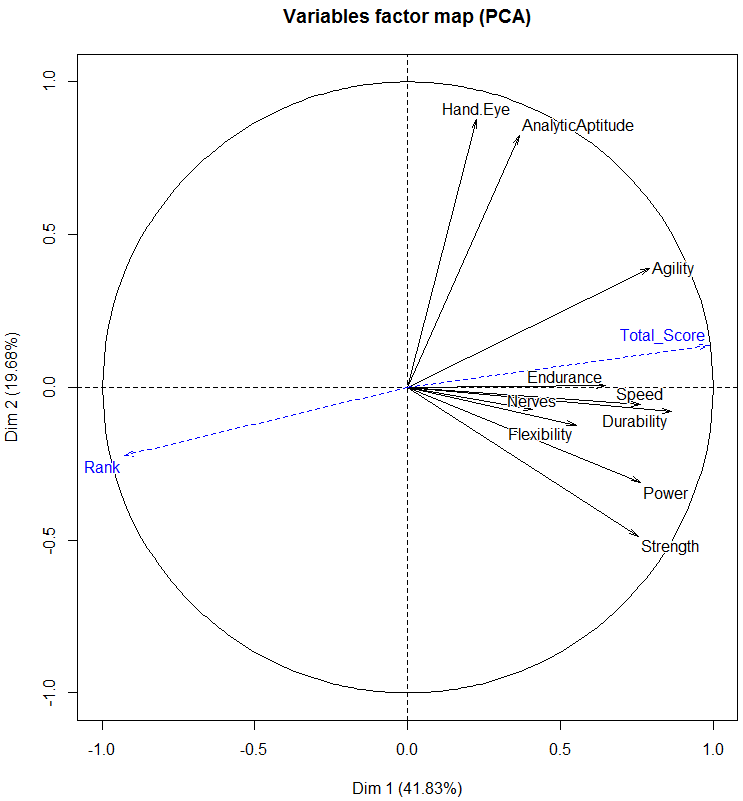
We again use FactoMineR PCA to explore the clusters .

> names(SportsDiff)

[1] "Endurance" "Strength" "Power" "Speed" "Agility"

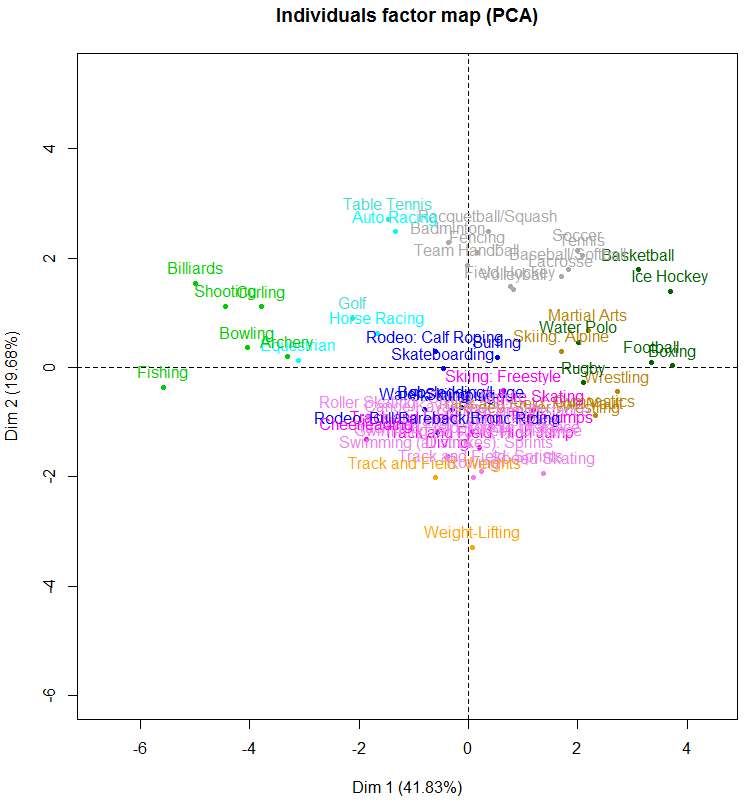
[6] "Flexibility" "Nerves" "Durability" "Hand.Eye" "AnalyticAptitude"

[11] "Total\_Score" "Rank"

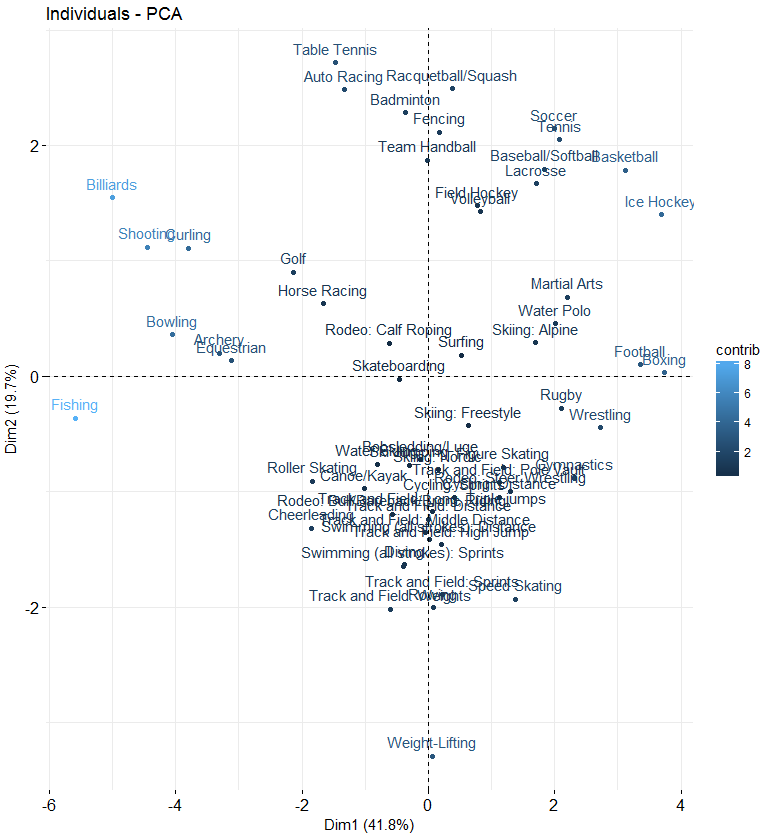
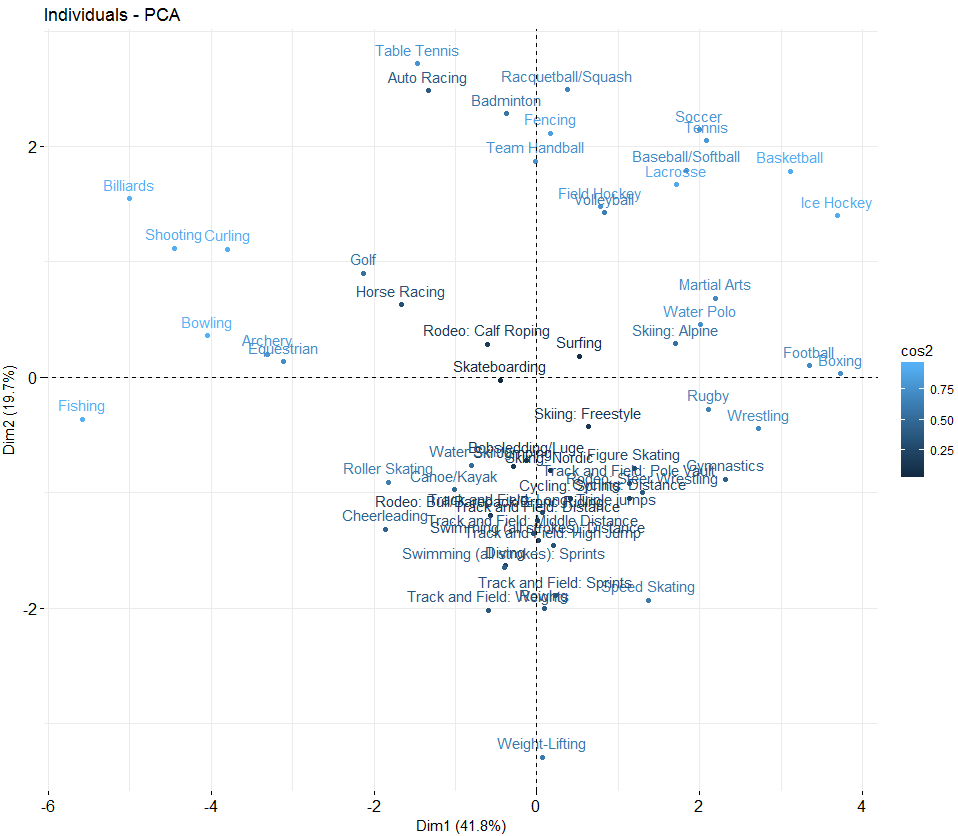
> sport.PCA = PCA(SportsDiff,quanti.sup=11:12)  


Here we have specified that both Total\_Score and Rank should be used as supplementary quantitative (i.e. continuous/numeric) random variables, meaning they will not be used in construction of the PC’s, but will displayed in any plots where variables are represented, e.g. biplots.

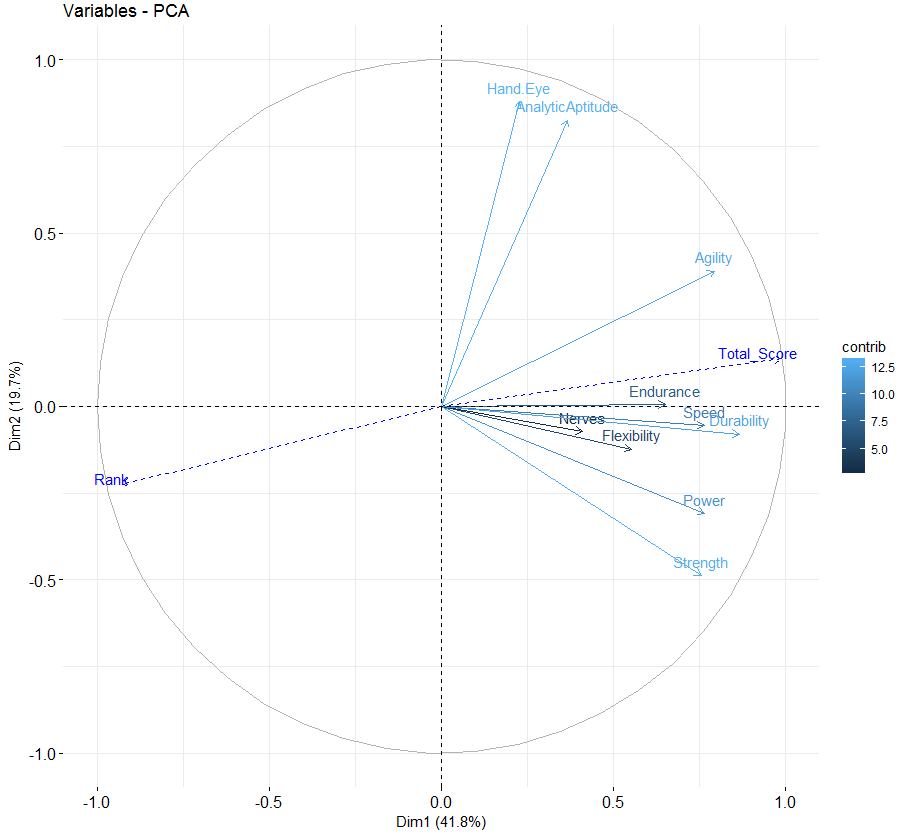
> plot(sport.PCA,col.ind=sport.kmeans$cluster+2)



> fviz\_pca\_ind(sport.PCA,col.ind="contrib")  
> fviz\_pca\_ind(sport.PCA,col.ind=”cos2”)

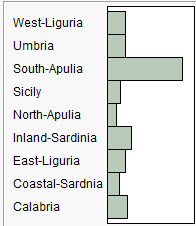
> fviz\_pca\_var(sport.PCA,col.var="contrib")



Example 5.7: Fatty Acid Content of Italian Olive Oils

Researchers are interested in characterizing differences in the fatty acid content of olive oils made from olives grown in different regions of Italy. There are two geographic classifications in these data. The first classification is nine individual growing areas in Italy (Area Name) – East Liguria, West Liguria, Umbria, North-Apulia, South-Apulia, Sicily, Coastal Sardinia, Inland-Sardinia, and Calabria. A broader classification is the growing region in Italy (Region Name) – Northern, Southern, and Sardinia. The map below should help in your understanding of where these areas/regions are located in Italy.





Puglia = Apulia

Sardegna = Sardinia

Sicilia = Sicily

The bar graph above shows the number of olive oils in these data from each area.

***The fatty acids measured are as follows:***

Palmitic

Palmitoleic

Stearic

Oleic

Linoleic

Eicosanoic

Linolenic

Molecular formulae taken from *Wikipedia*, so if these are wrong it is not my fault. I don’t understand how small differences in the number of carbon and hydrogen molecules make distinct fatty acids. Chemistry is weird!

We begin this example by using k-means clustering with the Italian olive oils in these data.

> names(Olives)

[1] "Region.name" "Area.name" "Region" "Area" "palmitic" "palmitoleic"

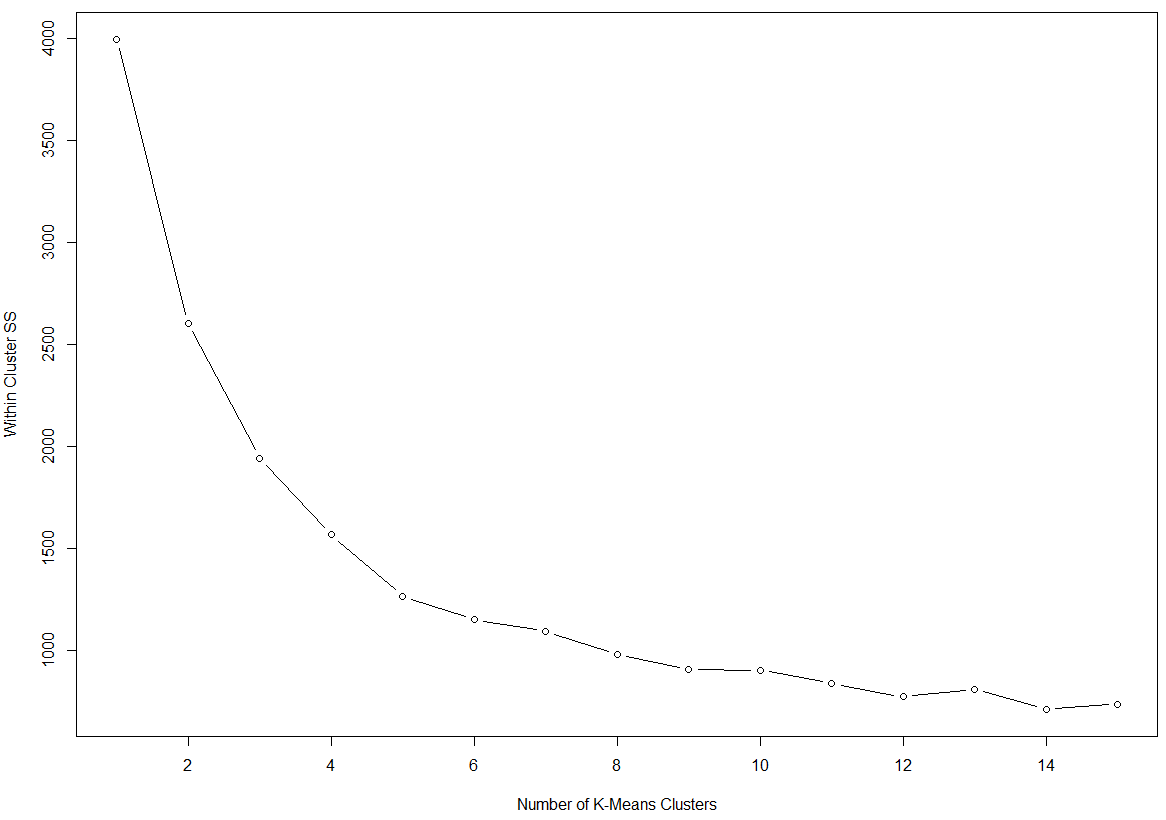
[7] "strearic" "oleic" "linoleic" "eicosanoic" "linolenic" "eicosenoic"  
  
> olive.mat = Olives[,5:11]

> olive.scale = scale(olive.mat 🡨scaling numeric variables when clustering   
 is critical when using distance based clustering   
 methods.

> wss = rep(0,15)

> for (i in 1:15) wss[i] = sum(kmeans(olive.scale,centers=i)$withinss)

> plot(1:15,wss,type="b",xlab="Number of K-Means Clusters",ylab="Within   
 Cluster SS")



The use of 7 – 10 clusters appears reasonable.   
  
Remember there are actually data from 9 growing areas in Italy.

> fit = kmeans(olive.scale,9) 🡨 use k = 9 clusters

> aggregate(olive.mat,by=list(fit$cluster),FUN=mean)🡨 more efficient way to obtain   
 group centers in the original scale.

Group.1 palmitic palmitoleic strearic oleic linoleic eicosanoic linolenic

1 1 1064.746 104.06780 256.1017 7698.644 861.8644 4.915254 9.491525

2 2 1438.226 183.41935 256.5806 6780.258 1177.9032 41.258065 69.322581

3 3 1080.564 73.29091 264.7455 7723.418 709.0909 41.654545 74.836364

4 4 1587.240 238.40000 210.3200 6563.680 1270.2000 32.880000 61.280000

5 5 1111.347 96.74490 226.1837 7268.020 1196.5306 27.091837 73.173469

6 6 1349.705 170.33333 219.5000 7058.103 1056.1410 34.192308 62.833333

7 7 1095.133 65.76000 203.5200 7914.027 618.6400 30.573333 47.013333

8 8 1290.538 113.83077 260.9846 7320.723 824.0154 46.246154 67.938462

9 9 1370.116 176.87209 192.0116 6939.942 1199.3488 34.151163 56.523256  
  
 Group centroids

> table(fit$cluster,Olives$Area)

1 2 3 4 5 6 7 8 9

1 0 0 0 0 0 0 9 50 0

2 0 1 25 5 0 0 0 0 0

3 15 0 0 15 0 0 24 0 1

4 0 0 25 0 0 0 0 0 0

5 0 0 0 0 65 33 0 0 0

6 0 2 70 6 0 0 0 0 0

7 9 2 0 0 0 0 14 0 50

8 1 51 0 10 0 0 3 0 0

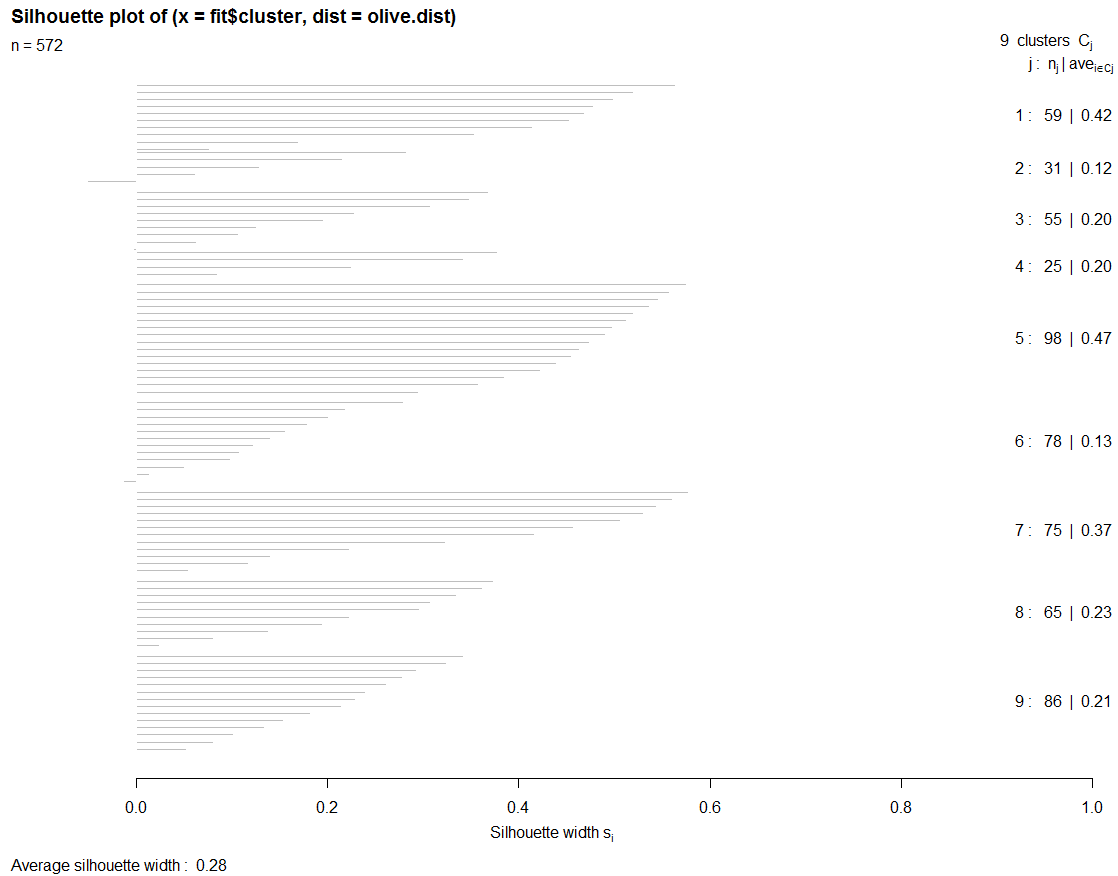
9 0 0 86 0 0 0 0 0 0

The clusters are fairly homogenous in terms of what growing areas the olive oils are from.

> olive.dist = dist(olive.scale)

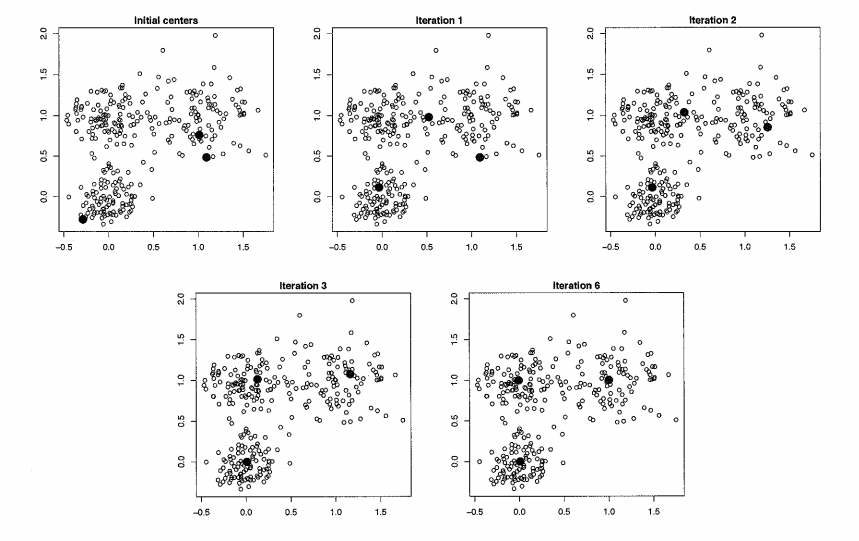
> si = silhouette(fit$cluster,olive.dist)

> plot(si)



Partitioning Around Medoids (PAM)

The partitioning around medoids (PAM) algorithm is similar to k-means but uses medoids (representative observations) rather than centroids (i.e. cluster means) to determine clusters. It takes either a data matrix or a dissimilarity matrix as an argument. The algorithm computes *k* representative objects, called medoids, which together with the distances determine the clustering. Each object in the data set is then assigned to the cluster corresponding to the nearest medoid. That is, object *i* is put into the cluster it is nearest to in terms of the distance to the *k* cluster medoids. After the initially assignment of all objects to the randomly chosen medoids, new medoids may be chosen and then assignment of all objects that are not medoids is done again. The process repeats until convergence.



PAM clustering is implemented by the pam function in the cluster library and there is a nice function for determining the “optimal” *k* value called pamk in the fpc library.

Below I illustrate the use of PAM clustering with the Italian olive oils data set.

> library(cluster)

> library(fpc) 🡨 needs to be installed from CRAN

> olive.mat = scale(Olives[,5:11]) 🡨 not using eicosenoic acid which is known to have errors.

> fits = pamk(olive.mat,krange=1:15,diss=F,criterion="asw",critout=T)

1 clusters 0

2 clusters 0.3120916

3 clusters 0.2831431

4 clusters 0.3273392

5 clusters 0.3703581

6 clusters 0.3585159

7 clusters 0.3342859

8 clusters 0.3283578

9 clusters 0.2734017

10 clusters 0.3085691

11 clusters 0.2757455

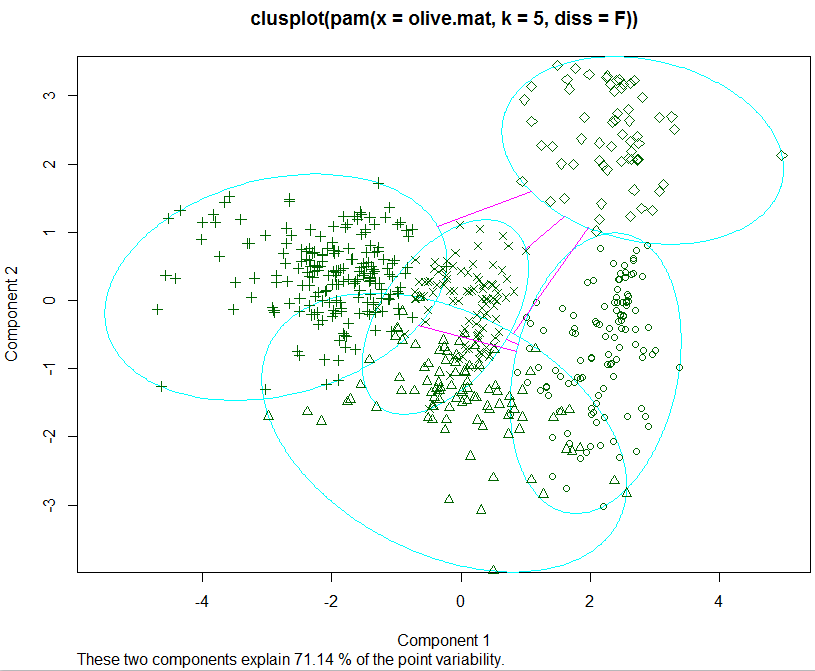
12 clusters 0.2646717

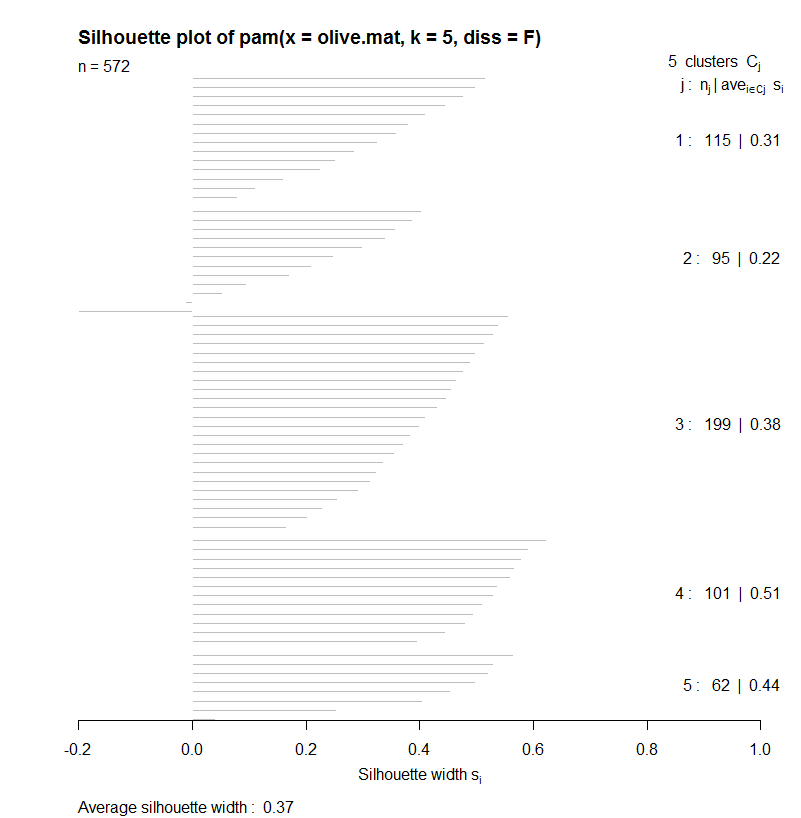
13 clusters 0.2591018

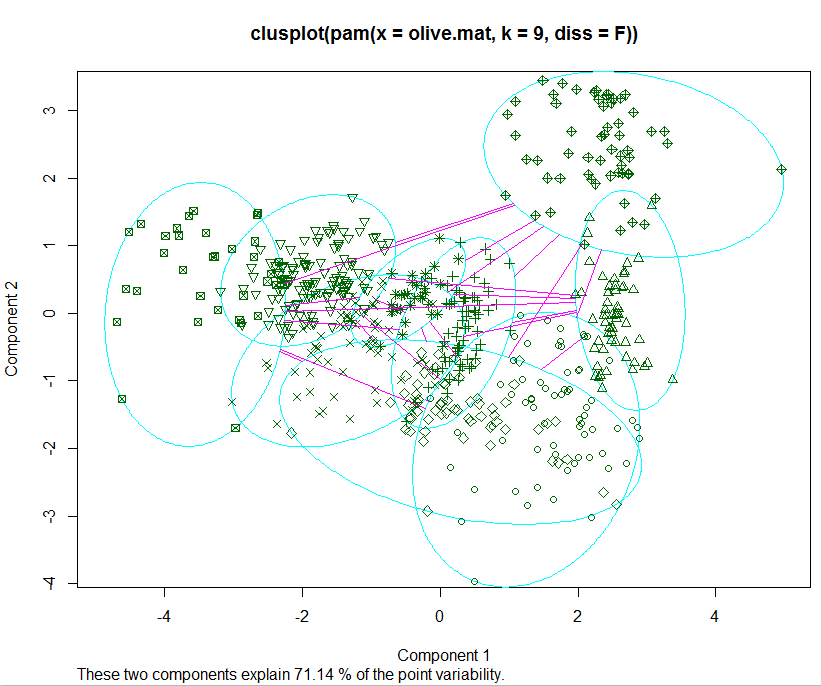
14 clusters 0.23787

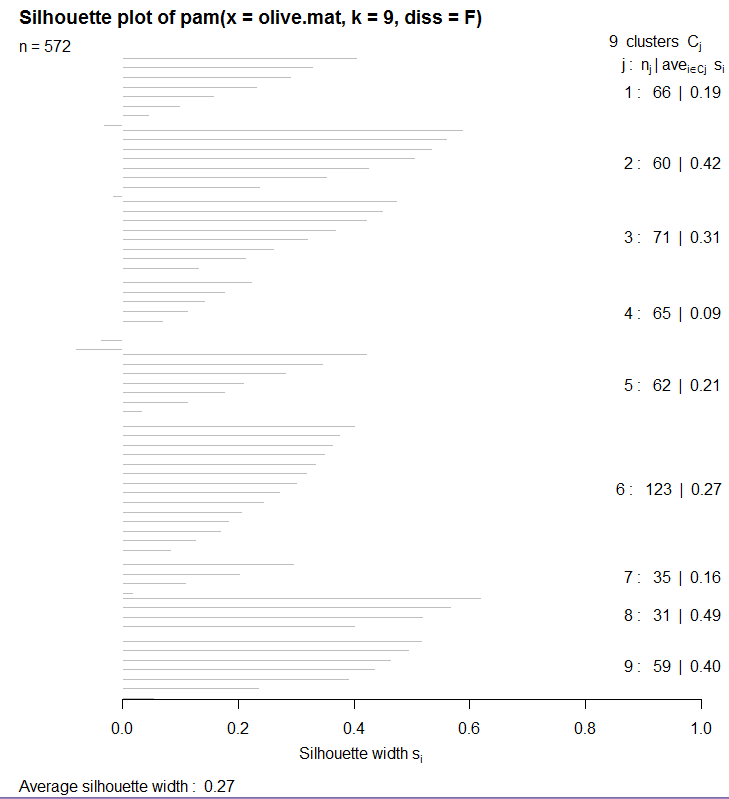
15 clusters 0.2406563   
  
> fit5 = pam(olive.mat,diss=F,k=5)

> plot(fit5)





> fit9 = pam(olive.mat,diss=F,k=9)  




> table(Olives$Area,fit9$clustering)

1 2 3 4 5 6 7 8 9

1 20 3 2 0 0 0 0 0 0

2 4 0 0 9 43 0 0 0 0

3 0 0 2 47 1 122 34 0 0

4 7 0 0 9 18 1 1 0 0

5 0 0 65 0 0 0 0 0 0

6 0 0 2 0 0 0 0 31 0

7 32 9 0 0 0 0 0 0 9

8 0 0 0 0 0 0 0 0 50

9 3 48 0 0 0 0 0 0 0

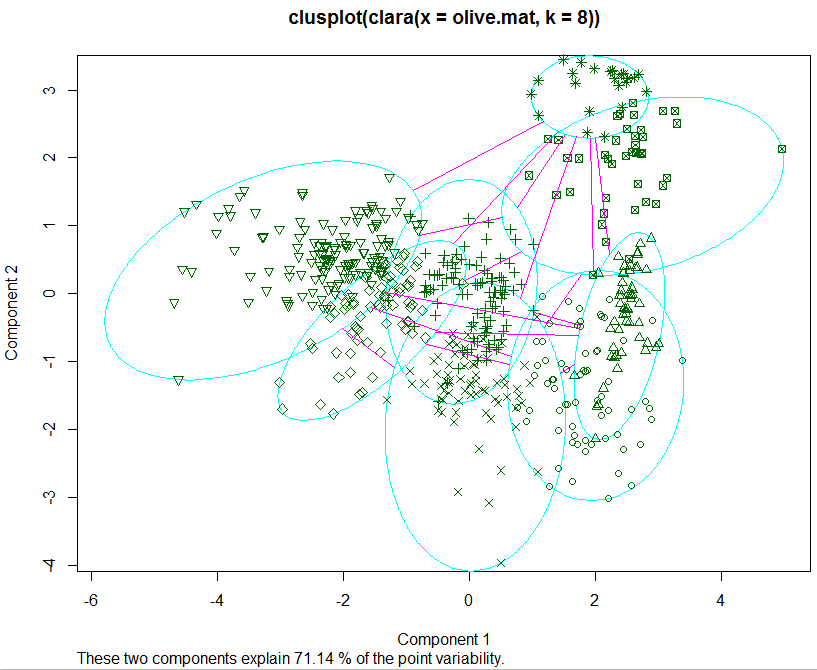
Again the clustering is strongly related to the nine growing areas, but not perfectly.

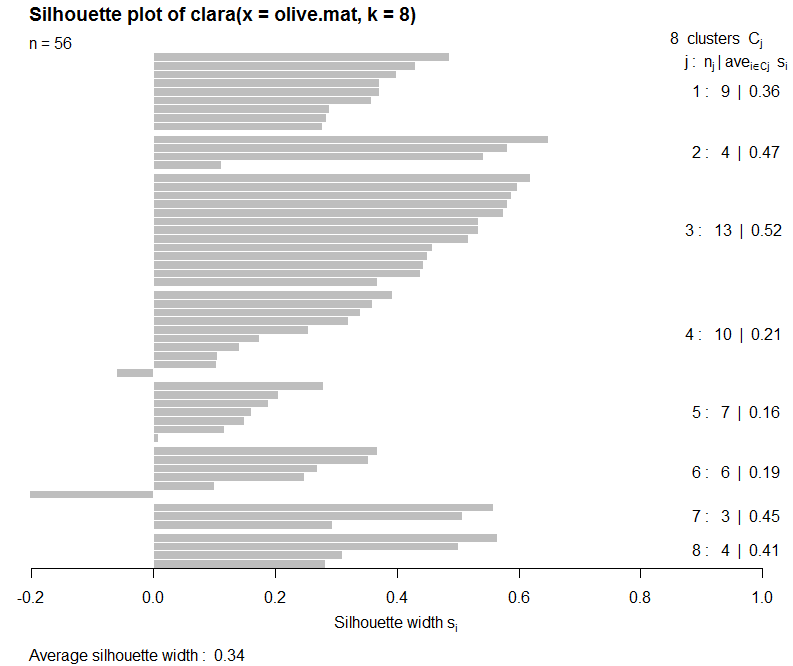
CLARA

CLARA stands for Clustering LARge Applications (large n). A random subset of the original observations is taken and each sub-dataset is partitioned into *k* clusters using the same algorithm as in PAM. Once k representative objects have been selected from the sub-dataset, each observation of the entire dataset is assigned to the nearest medoid. So basically CLARA performs PAM on a subset of the entire data set and then clusters the observations not chosen in the random subset to the clusters determined using PAM on the subset.

> fit8 = clara(olive.mat,8)

> plot(silhouette(fit8))





3-D Multidimensional Scaling (labels = clusters from CLARA, color = growing area)

